Machine Learning applied to Multifrequency Data in Astrophysics: Blazar Classification

B. Arsioli 1,2 and P. Dedin.1

1Gleb Wataghin Physics Institute, Unicamp, Rua S. Buarque de Holanda 777, 13083-859, Campinas, BR
2ICRANet, Piazza della Repubblica 10, 1-65122, Pescara, IT

Accepted XXX. Received yyy; in original form ZZZ

ABSTRACT

The study of machine learning (ML) techniques for the autonomous classification of astrophysical sources is of great interest, and we explore its applications in the context of a multifrequency data-frame. We test the use of supervised ML to classify blazars according to its synchrotron peak energy, either lower or higher than $10^{15}\;\text{Hz}$. We select a blazar sample with 4178 objects labelled as 1279 high synchrotron peak sources (HSP: $\nu$-peak $> 10^{15}\;\text{Hz}$) and 2899 low synchrotron peak sources (LSP: $\nu$-peak $< 10^{15}\;\text{Hz}$). A set of multifrequency input features were defined to represent each source, that includes spectral slopes ($\alpha_{\nu_1,\nu_2}$) between the radio, infra-red, optical, and X-ray bands, also considering IR colours. We describe the optimisation of five ML algorithms: Random Forests (RF), Support Vector Machine (SVM), K-Nearest Neighbours (KNN), Gaussian Naive Bayes (GNB) and the Ludwig auto-ML framework, and report on accurate models for blazar classification into LSP and HSP. In our particular case, the SVM algorithm had the best performance, reaching 93% of balanced-accuracy. A joint-feature permutation test revealed that the spectral slopes alpha-radio-IR and alpha-radio-optical are the most relevant for the ML modelling, followed by the IR colours. This work shows that ML algorithms can distinguish multifrequency spectral characteristics and handle the classification task of blazars into LSPs and HSPs, and is a hint for the potential use of ML algorithms for the autonomous determination of broadband spectral parameters (as the synchrotron $\nu$-peak), or even to search for new blazars in all-sky databases.

Key words: methods: data analysis – methods: statistical – galaxies: active – radiation mechanisms: non-thermal

1 INTRODUCTION

The autonomous classification of astrophysical sources is a challenge faced by modern astronomy, given the growing scale of its databases. High sensitivity detectors have opened a window for deep-sky searches able to reach distant and faint sources that are far too numerous to handle case-by-case. The study of efficient labelling and selection schemes via algorithmic modelling (Breiman 2001a) has become vital to explore big datasets and to help deliver refined scientific products. Consider, for example, the upcoming deep radio survey -i.e., the Evolutionary Map of the Universe (EMU Hopkins et al. 2015)- that is one of the leading science objectives from the Australian Square Kilometre Array Pathfinder (ASKAP). The ASKAP-EMU data-base will be crucial to search for early AGNs (Amarantidis et al. 2019) and is expected to identify nearly 75 million point sources down to 10 $\mu$Jy (to compare to ~2.5 million from NVSS (Condon et al. 1998), our deepest radio survey up to date). In this context, the classification of astrophysical objects will widely profit from machine learning techniques, which are ready to implement and open to fine-tuning optimisation.

We use multifrequency spectral data (from radio up to X-rays) to train, test, and compare several machine learning (ML) models applied to the classification of blazars according to the energy associated with its synchrotron peak. We aim to show that ML codes can classify blazars based on their multifrequency spectral characteristics, setting the ground for future works that aim to search for new blazars within multifrequency databases.

Blazars are a rare type of active galactic nuclei (AGN) known for its multifrequency emission over the entire electromagnetic spectrum and characterised by swift spectral variability. The central engines from AGNs are powered by matter accretion into supermassive black holes, which produce jets of relativistic charged particles (Padovani et al. 2017). When those jets point close to our line of sight, the sources are called blazars, and the observer perceives a relatively bright object due to relativistic boosting effects. The propagation of relativistic particles along the magnetised jets...
originates a non-thermal component which can cover many decades in energy, from radio up to TeV gamma-rays.

The presence of two non-thermal bumps in the spectral energy distribution (SED) of blazars (Fig. 1) originates from synchrotron and inverse Compton (IC) emission processes due to relativistic electrons moving through the jet’s magnetic field lines (Giommi & Padovani 2015; Giommi et al. 2012). Moreover, the thermal emission from the host galaxy and the accretion disk are both crucial elements to describe the overall spectrum, and a mixture of thermal and non-thermal components are at play (Urry & Padovani 1995), hindering autonomous identification and classification efforts.

A variety of selection criteria are proposed in the literature to search and identify new blazars (Arsioli et al. 2015; Chang et al. 2017; Di Mauro et al. 2014; D’Abrusco et al. 2012; Massaro et al. 2015) and are mostly based on the definition of qualitative boundaries in colour-colour planes, or by defining selection rules based on multifrequency spectral slopes. While one tries to simplify the selection rules and make it more tangible to comprehend, there is usually a lack of efficiency involved, meaning the final selection gets compromised with spurious sources and human intervention (as case-by-case evaluation) is necessary to clean the final sample.

ML algorithms are versatile in dealing with selection and classification problems in a multi-parameter space, where numerous input features are available (Ball & Brunner 2010; Borne 2009). A good example of challenging astrophysical label-problem is to disentangle ‘stars’ from ‘galaxies’ in optical observations. Common non-ML strategies used to handle the star-galaxy separation task involve colour-colour and proper-motion selection schemes (Windhorst et al. 2011). Alternatively, Bai et al. (2019) has shown that multifrequency data from optical and IR, together with ML techniques, can largely improve the accuracy to label stars, galaxies and quasars when compared to single-band classification methods.

In the case of blazars, the classification follows two main roads. One sole based on the characteristics of the optical spectra, and another based on the energy (frequency) associated with the synchrotron peak component. For the optical classification, blazars can be divides in BL Lacs (when the spectral data are dominated by the jet’s non-thermal component, therefore featureless), Flat Spectra Radio Quasar (FSRQ, when the optical spectra are dominated by thermal components from the AGN’s core, showing strong and broad emission lines), and for cases where such optical features are not clear or mixed (with weak or diluted emission lines within a relatively strong non-thermal component), the term uncertain/transitional blazar is used (Massaro et al. 2009, 2015). For the energy-based classification of blazars, the definition of families considers the synchrotron peak frequency. Accordingly, blazars are called as low, intermediate, high, and extreme synchrotron peak sources: LSP for \( \nu_{\text{peak}} < 10^{14} \text{Hz} \), ISP for \( 10^{14} \nu_{\text{peak}} < 10^{15} \text{Hz} \), HSP for \( 10^{15} \nu_{\text{peak}} < 10^{17} \text{Hz} \), and ESP for \( \nu_{\text{peak}} > 10^{17} \text{Hz} \) (Padovani & Giommi 1995; Abdou et al. 2010; Ghisellini 1999).

Both classification schemes demand a substantial human intervention to treat objects case-by-case, either to inspect optical spectral data to decide in between BL Lacs, FSRQ and Uncertain types, or to fit the non-thermal component and measure the synchrotron peak frequency. For the case of optical classification, Kang et al. (2019) attempts to use ML models to improve the labelling of uncertain blazars into BL Lacs or FSRQs, and predicting an FSRQ:BL Lac fraction of 1:3 within the uncertain blazar subsample, in accordance to numerous works (Chiaro et al. 2016; Massaro et al. 2016; Lefaucheur & Pita 2017; Yi et al. 2017).

Here we use an ML algorithmic approach to explore aspects of the energy-based classification of blazars, and train models to perform the task of labelling HSP and LSP sources (\( \nu_{\text{peak}} > 10^{15} \text{Hz} \) and \( \nu_{\text{peak}} < 10^{15} \text{Hz} \), respectively) only based on multifrequency information (fluxes and derived spectral slopes). Our long term goal, however, is to inspect and identify strategies to search for new blazars within large-scale astrophysical databases.

## 2 METHODOLOGY

There are currently twain blazar catalogues, namely the 5BZcat (Massaro et al. 2015) with 3561 objects, and the 3HSP (Chang et al. 2019) with 2013 objects (Note: 657 3HSPs are already listed in 5BZcat). The 5BZcat and 3HSP sum a total of 4915 unique sources which are either confirmed blazars, or blazar-candidates and are well characterised from radio to the X-ray band. Moreover, a large fraction has counterparts in the latest \( \gamma \)-ray catalogues, the 4FGL and 2BIGB (The Fermi-LAT collaboration 2019; Arsioli et al. 2020). In Section 3, we describe the use of high-accuracy labelled data from the 5BZcat and 3HSP catalogues to select a robust blazar sample, as well as the multifrequency cross-matching to build the ML data-frame. In Section 3.2, we use multifrequency fluxes to define spectral slopes between channels and apply statistical tests to select features for training our ML models.

In Section 4 the ML algorithms: RF, SVM, KNN, GNB, and Ludwig are briefly described, as well as the main evaluation metrics used in this work, and the re-sampling process used for the model’s validation process. In Section 5, there is a description of the optimisation for the Random Forest (RF), Support Vector Machine (SVM), K-Nearest-Neighbours (KNN), and Gaussian Naive Bayes (GBN) algorithms (Breiman 2001c; Cortes & Vapnik 1995; Goldberger et al. 2004; McCallum & Ngiam 1998). The optimisation is carried by scanning over the parameter space relevant for each algorithm using the SciKit learn libraries in Python (Pedregosa et al. 2011). We also test the performance of Ludwig, a deep-learning (auto-ML) tool released by the Uber team, but this time letting the tool auto-adjust its parameters (without user fine-tuning) to compare to our optimised models.

In Section 5.2, the level of balanced and absolute accuracy, precision and recall are used to evaluate and compare the performance of all ML models. We take into account of statistical fluctuations and report on mean output values as derived from hundreds of random re-samples. We track the metrics standard deviation as a direct way to evaluate the stability and robustness of each algorithm. In Section 5.3, we evaluate the importance of each input feature for optimised models, applying the so-called permutation test (a column associate to a given feature is shuffled, and the impact over the model accuracy is measured). The correlation between features is taken into account, and joint-feature permutation tests (shuffle of multiple features at once) are also described. In Section 5.4 we evaluate the performance of the optimised algorithms by tracking the misclassification-rate, which consists in counting the number of times an object is misclassified (by at least one algorithm) when considering 1000 re-samples. This way, we can identify and flag the most ambiguous cases.

---

1 The term ‘confirmed blazar’ refers to optically identified sources; therefore, an optical spectrum is available. All 5BZcat sources are confirmed blazars. The term ‘blazar-candidate’ refers to sources that lack optical identification. The 3HSP catalogue includes confirmed and blazar-candidates.
3 THE ML BLAZAR SAMPLE

The need to combine multifrequency databases is a known challenge in astrophysics, and currently, there is no autonomous solution for such a problem. To fuse databases is one of the most significant issues in the fields of Machine Learning and Artificial Intelligence as of today, and the building of data-frames is a fundamental step where substantial human effort is indeed needed (involving data handling, querying, cross-match, and fusion). There are ongoing and successful efforts to solve the data integration problem in astrophysics, and the Virtual Observatory (VO Becciani et al. 2010) is a sound example. Alternatively, there is software in place as Top-Cat (Taylor 2005) that can handle big data-sets in astrophysics, offering numerous cross-match and visualisation tools.

To build our data-frame we add a categorical attribute called ‘blazar-type’ to flag sources either as HSP+ESP (flag 1) or LSP+ISP (flag 0). The 5BZcat is the largest catalogue of optically identified (confirmed) blazars but has no synchrotron peak (nu_peak) associated with its sources. The categorical labels available from 5BZcat rely on optical classification: BZB for BL Lac, BZQ for FSRQs, and BZU for uncertain/transitional sources. To label 5BZcat sources as HSP or LSP, we use information from the 3HSP catalogue. The 3HSP Chang et al. (2019) is built over the search for HSP sources taking into account multifrequency selection rules together with a direct search within the 5BZcat and 3FHL (The Fermi-LAT Collaboration 2017) catalogues. Therefore, any object from 5BZcat that has no counterpart in 3HSP is considered an LSP (flag 0), and we label all 5BZcat-3HSP as HSP (flag 1). We use TopCat to cross-match the 5BZcat and 3HSP with 30″ radius (30 arc-seconds) and get 658 5BZcat sources classified as HSP+ESP (HSP: flag 1) and 2903 classified as LSP-ISP (LSP: flag 0).

At this stage, a blazar sample only based on 5BZcat sources is relatively unbalanced, because there are approximately 4.41 × more LSps than HSPs. There are many possible strategies to remedy the unbalance. During the ML model training, some algorithms allow to increase the weight associated to the less frequent class (the HSPs) or even generate new HSPs based on population properties of the HSP subsample (with the SMOTE technique, Chawla et al. 2002). In any case, the best approach to reduce the unbalance is to add extra real HSP sources to the sample.

The 5BZcat lists only 657 of the 2013 3HSP sources. Although the expected level of contamination for the 1356 3HSP-out-of-5BZcat is low (Chang et al. 2019), we only select cases that have a confirmed γ-ray counterpart from the 4FGL (The Fermi-LAT collaboration 2019) or the 2BIGB (Arsioli et al. 2020) catalogues. This selection translates into 627 additional HSPs: 482 that have a γ-ray counterpart in 4FGL and another 145 in 2BIGB. From those, 174 lack optical identification.

We rely on the γ-ray detection as a proxy to assure the blazar nature of the additional HSPs, keeping a high degree of purity in the data-frame. Finally, we define the ML-Blazar sample as the sum of the entire 5BZcat (3561 sources) with the γ-ray detected 3HSP-out-of-5BZcat (627 sources). At this stage, the ML-Blazar sample has a total of 4188 sources, classified as 1285 HSPs and 2903 LSPs, which improves the 5BZcat LSP sample has a total of 4188 sources, classified as 1285 HSPs and 3HSP-out-of-5BZcat (627 sources). At this stage, the ML-Blazar sample becomes balanced to all classes.

We read the radio fluxes reported in the 5BZcat and 3HSP catalogues, which comprises measurements from five different surveys: FIRST (1.4 GHz), NVSS (1.4 GHz), SUMSS (0.843 GHz), PMN (4.8 GHz) and TAPMN (4.85 GHz) (Helfand et al. 2015; Condon et al. 1998; Mroczka et al. 2003; Wright et al. 1994; McConnell et al. 2012) keeping track of the frequency νradio [Hz] associated to each measurement. Within the ML-Blazar sample, 29 sources are not detected in any radio survey. Nevertheless, those sources constitute robust blazars, well-characterised and detected in IR, optical, UV, X-rays, and most of them (25) are also detected in γ-rays with Fermi-LAT (4FGL and 2BIGB). Considering that NVSS and SUMSS have surveyed the entire radio-sky, we include upper limit (UL) values for the radio fluxes (2.5 mJy and 4.15 mJy, respectively)\(^3\). We add a categorical attribute to our data-frame called ‘radio-flag’: ‘0’ to flag sources already detected in radio, and ‘1’ for the non-detections (with UL values).

Table 1. Spectral information about the cross-match between the ML-Blazar sources with four optical catalogues (in top-down priority order).

| Catalog      | Channel name | ν × 10^−14 [Hz] | N matches |
|--------------|--------------|-----------------|-----------|
| GAIA DR2     | Gp           | 3.761           | 3840      |
| SDSS DR12    | r            | 4.862           | 2073      |
| PanStars DR1 | R            | 4.823           | 3523      |
| USNO B.1     | R            | 4.454           | 3948      |

1 SMOTE: Synthetic Minority Over-sampling Technique, is an algorithm that identifies the minority class and randomly select members of it, look into the k-Nearest Neighbours (n-neighbours=5) accounting for the available features, and produce new members of the minority class. The final sample becomes balanced to all classes.

3 Following Condon et al. (1998), the completeness limit for NVSS is about 2.5 mJy (2.5 × 10^-26 erg/cm^2/s/Hz) at 1.4 GHz, and similar for SUMSS (Mroczka et al. 2003) which converts to 4.15 mJy at 0.843 GHz.

Machine Learning & Multifrequency Astrophysical Data
follow the WISE team instructions\footnote{For instructions to convert the observed WISE magnitudes to erg/cm$^2$/s, see http://wise2.ipac.caltech.edu/docs/release/allsky/expsup/sec4_4h.html} to convert magnitude into flux density values [erg/cm$^2$/s], considering the case of an underlying point-sources with power-law spectra.

**Optical.** For the optical band, the ML-Blazar sample was cross-matched with four catalogues, including (in priority order): GAIA DR2, SDSS DR12, PanStars DR1, and USNO B.1 (Gaia Collaboration et al. 2018; Alam et al. 2015; Chambers et al. 2016; Monet et al. 2003), considering 3° radius. We keep only the best match within each optical database (Table 1) and concatenate all matches in a single table; that one has 13384 correspondences between the ML-Blazar sample with the four optical catalogues. All repeated sources were removed by running an internal cross-match with TopCat, keeping a single optical match for each ML-Blazar source, and considering the priority order for the optical catalogues. As a result, 4122 out of 4180 ML-Blazars (98.6%) have optical information assigned. Given that the 5BZcat also reports the Rmag ($r_w$), Optical (green), and X-ray (grey). The fluxes corresponding to each blazar were connected with yellow translucent lines to form a density view of the parameter space covered by the entire data-frame.

The ML-Blazar sample had no measured X-ray flux ($\sim$25%). Considering that RASS is the only all-sky survey available, and given that it covers the majority of the sky out of the galactic plane with flux-limit ranging from $10^{-1}$ to $10^{-3}$ erg/cm$^2$/s at 1 keV (flux density of $\sim 4-0.4 \times 10^{-10}$ erg/cm$^2$/s), we add an X-ray flux-limit of $5.0 \times 10^{-13}$ erg/cm$^2$/s ($\sim 2.0 \times 10^{-10}$ erg/cm$^2$/s) to all cases with no X-ray counterpart. To keep easy track of the UL value, we add the X-rayUL attribute with value 1 for UL, and 0 for measured flux.

### 3.2 Feature selection and Standardisation

Following the preprocessing stage, the ML-Blazar data-frame has six flux attributes ($f$: radio, IR-w1, IR-w2, IR-w3, optical, X-ray), and two main categorical flags (source-type and X-rayUL) associated to each source, summing a total of 8 attributes. To try and condensate multifrequency information into fewer parameters, we calculate spectral slopes $\alpha_{\nu_1 \nu_2}$ between bands according to:

$$
\alpha_{\nu_1 \nu_2} = \frac{\log(f_{\nu_1}/f_{\nu_2})}{\log(\nu_1/\nu_2)}
$$

The spectrum slopes are well representative of the energy distribution (SED shape) needed for our classification, and carry information from two bands into a single $\alpha$ parameter. We test the possibility to reduce the number of attributes necessary to train our ML model, by working with slopes instead of fluxes. However, at this stage, any combination of bands (into $\alpha$) could turn out relevant, so that we add another fifteen attributes to the data-frame for evaluation purposes: $\alpha_{\nu_1 \nu_2}$, $\alpha_{\nu_1 \nu_3}$, $\alpha_{\nu_2 \nu_3}$, $\alpha_{\nu_4 \nu_5}$, $\alpha_{\nu_6 \nu_7}$, $\alpha_{\nu_7 \nu_8}$, $\alpha_{\nu_9 \nu_10}$, $\alpha_{\nu_11 \nu_12}$, $\alpha_{\nu_12 \nu_13}$, $\alpha_{\nu_13 \nu_14}$, $\alpha_{\nu_14 \nu_15}$, and $\alpha_{\nu_15 \nu_16}$. By defining slopes between channels, we wish to retain information within a small number of attributes. We look forward to reducing the dimension of the problem by keeping only the ones that are more promising to separate between blazar classes.

We perform the standardisation (std) of all features, to represent them as normal distributions with 0.0 mean value and unitary standard deviation ($\sigma$). To do that, we take each input value, e.g. $\alpha_{\nu_1 \nu_2}$, subtract it by the feature-mean $(\langle \alpha_{\nu_1 \nu_2} \rangle)$, and divide by the

### Table 2. Cross-match radius for each X-ray catalogue, adapted according to the position-error $p_{\text{err}}$ reported for each source. Columns describe the catalogue name, the total number (N) of sources in each of them, the position error range $p_{\text{err}}$ (used to group sources before cross-matching with the ML-Blazar sample), and the fixed radius assumed for the cross-match.

| Catalogue            | N     | $p_{\text{err}}$ | radius |
|----------------------|-------|------------------|--------|
| 2RXS-RASS            | 135.181 | all-data         | 40"    |
| Swift-1SWXRT         | 84.979  | $0.5''$          | 0.1''  |
| Swift-XRTGRB         | 151.524 | all-data         | 5''    |
| XMM-DR8(2018)        | 531.454 | $0.5''$          | 0.1''  |
| XMM-SL2(2017)        | 29.393  | all-data         | 10''   |
| Chandra-V1.1         | 106.586 | all-data         | 0.1''  |

---

**Figure 1.** A superposed SED of all sources of the ML-Blazar data-frame. From left to right, the bands are: Radio (red), IR $w_1$ (cyan) $w_2$ (magenta) $w_3$ (blue), Optical (green), and X-ray (grey). The fluxes corresponding to each blazar were connected with yellow translucent lines to form a density view of the parameter space covered by the entire data-frame.
feature standard deviation $\sigma_{\alpha_{l\leq 5}}$, eq. 2. The mean and $\sigma$ values are calculated for the feature distribution considering all sources together (meaning: there is no need to separate HSP and LSP classes) and are listed in Table 3. Standardisation is important for the optimisation of the machine learning algorithms because models can misleadingly assign a large weight (importance) to a given feature, only because of its dominant absolute value. Therefore, the standardisation is commonly applied to level-out all features, so that all have the same weight when computing cost functions during the optimisation process.\(^5\)

\[
\log(f)_{\text{std}} = \frac{\log(f) - \langle \log(f) \rangle}{\sigma_{\log(f)}} = \frac{\alpha_{\alpha_{l\leq 5}} - \langle \alpha_{\alpha_{l\leq 5}} \rangle}{\sigma_{\alpha_{l\leq 5}}} \quad (2)
\]

A Kolmogorov-Smirnov test (KS, Hodges 1958) is applied to evaluate if the HSP and LSP samples are drawn from the same distribution (the null hypothesis $H_0$). The KS test statistic $D$ is the maximum absolute difference between the two cumulative distribution functions, and the corresponding two-sided p-value is the probability that we would see a $\geq D$ value, only by chance, given that $H_0$ is true. A two-sided p-value $< 0.05$ is usually interpreted as a denial of $H_0$, meaning that the LSP and HSP distributions are not drawn from the same population. Using the scipy.stats.ks-2samp library (Jones et al. 2001), the p-values obtained for all spectral slopes (Table 3) reject $H_0$ at $<1\%$ level, meaning that the distributions are not drawn from the same populations. In other words, all features $\alpha_{l\leq 5}$ hold unique information about the LSP and HSP classes. A collection of five slopes $\alpha_{rw1}$, $\alpha_{ro}$, $\alpha_{rx}$, $\alpha_{w1w2}$, $\alpha_{w1w3}$ enclose information along the radio to the X-ray band, with the largest KS $D$ values (all $> 0.7$) and the lowest $P$-values (all $\sim 0.0$). The largest $D$ values associated to nearly null p-values are a strong indication that those five slopes are the most relevant parameters to separate/classify blazars into LSP or HSP type.

We also apply a t-test for independent-samples to compare the mean values of a given feature and for each class. It measures how separable are the classes according to the means, and the absolute t-test value can be used to compare the features (given that all are standardised). According to the t-test, all features are distinct with respect to the mean, at high confidence level, p-values $< 1\%$; and the six best performing features (most separable according to the mean) are: $\alpha_{ro}$, $\alpha_{rw1}$, $\alpha_{w1w2}$, $\alpha_{w1w3}$, $\alpha_{rx}$, and $\alpha_{w2w3}$. Those enclose all the five slopes highlighted by the Kolmogorov-Smirnov test, therefore pointing to a similar conclusion.

From now on, we focus on these five spectral slopes ($\alpha$: $rw1$, $ro$, $rx$, $w2w1$, and $w3w2$) to train and optimise several machine learning algorithms while keeping a low number of features. The dimension of the problem reduces from seven input features (six flux density + one X-ray upper limit flag) to only five spectral slopes. Moreover, the statistical tests show that spectral slopes can separate blazar classes better than the fluxes alone (as seen by the results in table 3).

Fig. 3.2 shows the density distribution of HSPs and LSPs for the five selected alphas, as a primary assessment of how separable are LSP and HSP sources based on each feature. This rather qualitative view is complementary to the results reported in Table 3, which lists the Kolmogorov-Smirnov test statistics $D$ and the t-test statistics, applied to compared the LSP and HSP distributions.

\(^5\) Standardisation is especially relevant for algorithms which rely on the calculation of Euclidean distances between points in multidimensional space (Support Vector Machine, K-neighbours), or when gradient descent is used for optimisation in Neural Networks.

---

**Table 3. List of mean values and standard deviations associated with each feature, as used for its standardisation (eq. 2). From the centre to the right columns, the KS test statistic $D$ and the t-test statistics, with their corresponding p-values, to compare the feature’s distributions for HSP and LSP sources. The best performing features for the classification task are marked with *.

| Features          | Mean  | $\sigma$ | KS D | p-value | t-test | p-value |
|-------------------|-------|----------|------|---------|--------|---------|
| **Flux Density**  |       |          |      |         |        |         |
| log(F Radio)      | -23.98| 0.735    | 0.66 | 0.0     | 56.1   | 0.0     |
| log(F IR-w1)      | -26.15| 0.530    | 0.11 | 2E-10   | 4.7    | 2E-6    |
| log(F IR-w2)      | -26.07| 0.529    | 0.13 | 7E-15   | 5.9    | 4E-9    |
| log(F IR-w3)      | -25.82| 0.530    | 0.51 | 2E-205  | 27.8   | 4E-152  |
| log(F Opt)        | -26.63| 0.563    | 0.17 | 2E-24   | 10.9   | 5E-27   |
| log(F X-ray)      | -29.67| 0.466    | 0.46 | 7E-161  | 25.0   | 4E-120  |
| **Radio-IR**      |       |          |      |         |        |         |
| alpha-rw1*        | 0.448 | 0.170    | 0.71 | 0.0     | 62.3   | 0.0     |
| alpha-rw2         | 0.445 | 0.159    | 0.66 | 0.0     | 54.7   | 0.0     |
| alpha-rw3         | 0.429 | 0.145    | 0.51 | 3E-205  | 37.7   | 3E-263  |
| **Radio-O&X**     |       |          |      |         |        |         |
| alpha-ro*         | 0.484 | 0.162    | 0.71 | 0.0     | 64.1   | 0.0     |
| alpha-rx*         | 0.686 | 0.111    | 0.76 | 0.0     | 66.7   | 0.0     |
| **IR-colours**    |       |          |      |         |        |         |
| alpha-w2w1*       | 0.572 | 1.046    | 0.72 | 0.0     | 53.8   | 0.0     |
| alpha-w3w2        | 0.609 | 0.621    | 0.71 | 0.0     | 48.6   | 0.0     |
| alpha-w1w3*       | 0.600 | 0.664    | 0.77 | 0.0     | 58.1   | 0.0     |
| **IR-Optical**    |       |          |      |         |        |         |
| alpha-w1o         | 0.751 | 0.595    | 0.27 | 5E-58   | 9.4    | 1E-20   |
| alpha-w2o         | 0.722 | 0.527    | 0.43 | 4E-147  | 25.2   | 1E-129  |
| alpha-w3o         | 0.683 | 0.433    | 0.66 | 0.0     | 48.1   | 0.0     |
| **IR-Xray**       |       |          |      |         |        |         |
| alpha-w1x         | 0.751 | 0.595    | 0.24 | 2E-44   | 17.2   | 1E-62   |
| alpha-w2x         | 1.002 | 0.166    | 0.37 | 8E-108  | 26.4   | 7E-135  |
| alpha-w3x         | 0.961 | 0.164    | 0.56 | 2E-245  | 44.6   | 3E-315  |
| **Opt-Xray**      |       |          |      |         |        |         |
| alpha-ox          | 1.076 | 0.204    | 0.16 | 5E-20   | 11.5   | 4E-30   |

---

**4 THE MACHINE LEARNING ALGORITHMS AND METRICS**

Here we briefly describe the machine learning algorithms used in this work (Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbours (KNN), Gaussian Naïve Bayes (GNB), and Ladwig) and the configuration parameters which are available to optimise each of them. Following, we discuss the metrics used to evaluate the performance of the ML models, and the re-sampling strategy applied to validate and access the uncertainty associated with each of them.

**Random Forest (RF):** The Random Forest\(^6\) is an extension of the Decision Tree method. Decision Trees are supervised machine learning algorithms used for classification based on decision rules. These decisions are organised hierarchically in nodes, where a decision is made based on the feature that best separates the classes in that node using a given metric (either gini indices or entropy). The tree starts on the so-called root node, made by the feature with the best dividing metric for the whole data, and ends on leaves where there are no better divisions to be made. To work around the problem of biased Decision Trees, the Random Forest uses an ensemble of trees.

\(^6\) Scikit-learn RF documentation.
of trees, each one built from a random sample of the training set (bootstrap sample). Also, instead of using the best split among all the features, the choice is made among a random subset of features. Therefore, in this kind of algorithm, the most critical optimisation parameters are the number of trees (or estimators), the size of the features subset and the tree’s maximum depth. Finally, each tree of the Random Forest counts as a vote for the classification of each object, and the ensemble of votes defines the final classification.

Support Vector Machine (SVM): The Support Vector Machine classifier\(^7\) uses hyperplanes on the features space trying to best separate the objects among their classes. The algorithm maximises the distance between the hyperplane and the objects of any class, trying to minimise classification errors. One attribute of this kind of algorithm is the Kernel, that is a transformation from the features space to another space that better separate the classes. The most used Kernels are the linear, polynomial, radial basis function (RBF) and sigmoid. For all the SVM kernels there is the parameter called C, which is a penalisation for the wrong classification so that a high C forces a better and more precise classification, but it could also bias towards overfitting the training data. For the polynomial, RBF and sigmoid, there is also the gamma parameter, which is a scale factor for the computed distances in these kernels. For the polynomial, there is also its degree.

K-Nearest Neighbours (KNN): The K-Nearest Neighbours algorithm\(^8\) implements a voting system where the class of the nearest neighbours objects define the classification. In the simplest implementation, the algorithm computes the object class by the majority of votes where all the k nearest neighbours have the same weight. Another approach is to weight the neighbour vote by the inverse of its distance from the object. In both cases, by distance, we mean the Euclidean distance, which can be computed in multi-dimensional parameter space. Therefore, the main variables to play in this model are the number of neighbours used in the classification and the weight for them, uniform or radius dependent.

Gaussian Naive Bayes (GNB): The Naive Bayes\(^9\) classifiers use the Bayes Theorem from statistics to predict the class given the features of an object, making the “naive” assumption of conditional independence among the features to simplify the calculations. The result of this assumption is a simple formula:

\[
P(y|x_1, \ldots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i|y)
\]

where \(y\) is the class and \(x_i\) the \(i\)th feature of a given object. The Gaussian Naive Bayes algorithm is the one that assumes \(P(x_i|y)\) to follow a Gaussian distribution. Also, \(P(y)\), called prior probability, is the probability of each class in the dataset. To predict the class, the GNB algorithm chose \(y\) that maximise the probability that the object belongs to that class. Although this type of algorithm uses simplified assumptions, it produces good results in a variety of applications, besides being one of the lightest algorithms to run.

Ludwig: Ludwig is an open-access framework of deep learning, able to deal with classification-problems, and designed to be autonomous in fine-tuning its internal parameters to accomplish efficient and accurate classification of objects (Molino et al. 2019). Ludwig is made available by the Uber team\(^{10}\), and we use it to compare its automated modelling with our optimised results. This way, we access the power of Ludwig for science use cases, noting it is versatile and relatively easy to implement.

4.1 Statistical Analysis & Evaluation Metrics

One of the most common metrics used to evaluate the model response is the accuracy (accuracy_score, Pedregosa et al. 2011), defined as the fraction of objects correctly classified by the model (the number of true positives TP, divided by the total number of objects in the sample, \(n_{samples}\)) which gives us a performance response independent of class. To complement the analysis, two other metrics

---

\(^7\) Scikit-learn SVM documentation.
\(^8\) Scikit-learn KNN documentation.
\(^9\) Scikit-learn GNB documentation.
\(^{10}\) Ludwig on Github: https://github.com/uber/ludwig.
recall and precision are commonly considered for model evaluation; in a “per class” perspective, those represent the completeness of a class classified by the ML model and the efficiency of the classifier. The precision is defined as the number of true positives (TP) divided by the number of elements classified as belonging to the class (True Positives + False Positives: TP + FP); the recall is defined as the true positives (TP) divided by the number of elements in the class (True Positives + False Negatives: TP + FN) (classification_report, Pedregosa et al. 2011). The precision tells how clean the classification for each given class, and the recall tells the fraction of objects of a class that were classified correctly.

Another metric commonly used to evaluate the performance of categorical classifiers is the f1-score, which represents the weighted average between precision and recall for each class. The use of f1-score has its limitations since it gives equal weight to precision and recall, but it can be used to compare models in terms of the optimum balance between precision and recall.

However, for an imbalanced data set, both the accuracy and f1-score can not provide the full picture, because the metric gets biased towards the dominant class, i.e. the LSPs. To account for the imbalance we look for the scikit-learn implementation of the (balanced_accuracy_score, Brodersen et al. 2010) that correspond to the average of recall for each class; and the weighted-average f1-score,11 that calculates both the precision and recall metrics for each label, and weight the f1-score based on the number of objects in each class. All metrics listed were used to evaluate the models regarding the test samples:

\[
\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} = \frac{TP_{\text{all-classes}}}{n_{\text{samples}}}
\]

\[
\text{Bal-Accuracy} = \frac{1}{2} \left( \frac{TP_{\text{class-A}}}{n_{\text{class-A}}} + \frac{TP_{\text{class-B}}}{n_{\text{class-B}}} \right)
\]

\[
\text{Precision} = \frac{TP}{TP + FP}; \text{ perclass}
\]

\[
\text{Recall} = \frac{TP}{TP + FN}; \text{ perclass}
\]

\[
\text{f1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}; \text{ perclass}
\]

\[
\text{Average-Weight f1} = \frac{n_{\text{class-A}} \times f1_{\text{class-A}} + n_{\text{class-B}} \times f1_{\text{class-B}}}{n_{\text{samples}}}
\]

In the perspective of population studies in astrophysics, precision and recall translates respectively into selection-efficiency and sample-completeness. Both completeness and efficiency are highly valuable for astrophysics, but the trade-off between them may depend on the application (Ball & Brunner 2010). In case of searching for rare objects as blazars within large multifrequency databases, the most effective strategy would likely be to give priority to completeness (recall). Allowing lower efficiency—which leads to a certain degree of contamination—can be remedy latter by additional strategies, as cleaning the final database with hand-implemented cuts, or even by human intervention via case-by-case investigation of border-class cases.

4.2 Re-sampling & Validation

To analyse the results of the classifier, a commonly used tool in machine learning is the so-called K-fold cross-validation, where the data set is divided into K parts, with one of these used as the testing set and the rest used for training. The process is repeated for each K part of the testing set to calculate the average over the performances. Our approach is similar, but instead of pre-dividing the data set, we split it for training and testing using a pseudo-random algorithm and repeat this process 300 times, so that each time we have a randomly selected train and test sets. The final result is a distribution of the performance, where we can extract the mean and standard deviation (σ) associated with any metric. The degree of uncertainty of a given model will directly translate into low σ, signalling that the model is stable regarding randomly selected train and test samples, and up to the classification task for real applications.

---

11 About the balanced (weighted) f1-score, check https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html.
was performed again, and from that follows the results we report here\textsuperscript{12}. Fig. 3 (top) shows the mean balanced-accuracy (bal-acc) and its standard deviation as a function of the test sample size, for the RF, SVM, KNN, and GNB models. The mean bal-acc and $\sigma$ are calculated from 300 re-samples. From there we decide to split the ML-Blazar sample into 65% for training and 35% for testing, to keep the test uncertainty at $\sim$1\% level for all models while keeping the mean balanced-accuracy as high as possible, and depleting the training sample as less as possible.

We also tested if our data-frame is large enough concerning the total number of objects (sample size). Using a random split, we reduced the dataset to smaller samples and trained the ML algorithms. At first, the algorithms were train $\&$ test using the default configuration, and later the results were updated with the optimised versions (the ones described in Sec. 5.2). Fig. 3 shows the bal-acc and $\sigma$ scanning the dataset sizes from 70 to 4000, and considering 300 re-samples for each point. As seen, the mean bal-acc evolves from 88-90\% up to 91-93\%, and the $\sigma$ reduces from a $\sim$15\% band

\textsuperscript{12} This results in a recurrence problem, where at each iteration the configuration converges to the best one. However, for this specific test, we obtained just minor corrections in the first iteration, and the selected train:test ratio remains the same, 0.65:0.35.

5 DISCUSSION AND RESULTS

Here we walk through the steps of model optimisation for the RF, SVM, KNN, and GNB algorithms. Each ML algorithm has a different space of fitting parameters to probe, and we scan over them to select the best configuration focusing on the balanced-accuracy metric. For the Ludwig framework, we only feed the data and collect results, with no optimisation other than its auto-ML functionality.

5.1 optimising the train and test sample size

Before any attempt to optimise the ML models, we tried to understand which is the best splitting between train and test samples, and set the test size to vary between 10\% to 90\% of the total sample size. Initially, the ML algorithms run over default configuration (as specified for each algorithm at sci-kit learn library, Pedregosa \textit{et al.} 2011) and using the balanced-accuracy as performance criteria. Later on, after we optimise the ML algorithms, the evaluation was performed again, and from that follows the results we report here. The mean balanced-accuracy (bal-acc) and its standard deviation as a function of the test sample size, for the RF, SVM, KNN, and GNB models. The mean bal-acc and $\sigma$ are calculated from 300 re-samples. From there we decide to split the ML-Blazar sample into 65\% for training and 35\% for testing, to keep the test uncertainty at $\sim$1\% level for all models while keeping the mean balanced-accuracy as high as possible, and depleting the training sample as less as possible.

We also tested if our data-frame is large enough concerning the total number of objects (sample size). Using a random split, we reduced the dataset to smaller samples and trained the ML algorithms. At first, the algorithms were train $\&$ test using the default configuration, and later the results were updated with the optimised versions (the ones described in Sec. 5.2). Fig. 3 shows the bal-acc and $\sigma$ scanning the dataset sizes from 70 to 4000, and considering 300 re-samples for each point. As seen, the mean bal-acc evolves from 88-90\% up to 91-93\%, and the $\sigma$ reduces from a $\sim$15\% band.
down to ~1% band. Although the gains in bal-acc seem relatively small, the $\sigma$ shrinks drastically as the sample size grows, meaning that the uncertainty associated to each trained model is lower (as expected), signalling that the modelling is increasingly stable and robust. By this analysis, we assure that our data-frame is large enough to produce models with small uncertainty and a high level of balanced-accuracy.

### 5.2 Fine-tuning the ML Models

Here we describe the optimisation steps for each ML algorithms, scanning over the available parameters to fine-tune the models. We look forward to maximising the performance of the mean balanced-accuracy and report on the optimal setup conditions to meet that goal. We adopt the re-sampling as part of the uncertainty analysis and model validation (Kelling et al. 2009), which showed to be a rather compute-intensive task. Our re-sampling analysis consists of 300 full train and test evaluations, running over randomly selected train & test samples, to access the standard deviation of all target metric (e.g. the balanced-accuracy).

The choice of the best performing model, however, depends on the starting goals. As an example, to maximise the LSP or HSP recall (completeness) without caring for the contamination of the sample, one could use different optimisation parameters. To illustrate those possibilities, in addition to the balanced-accuracy dependence, we show the precision, recall, and balanced-f1 scores.

**Random Forest (RF):** The main parameters we optimise were: the number of features, the number of trees (or estimators), and the tree’s maximum depth. We balance the training sample by assigning a larger weight to the less frequent class (the HSPs); setting the option `class_weight=‘balanced’`. The number of features was scanned from 1 to 5, showing better performance at 1; the number of trees was scanned from 10 to 1000, performing best at 250; and the maximum depth was scanned from 1 to 15, performing best at 5, as shown in Fig. 4. Those results emerge from an iterative procedure, starting with default values, and followed by fine-tuning, reaching an optimum balanced-accuracy of 0.926±0.006.

**Support Vector Machine (SVM):** We scan over the cost (C) value, for the range between 0.003 to 500, and each of the four kernels: rbf, linear, polynomial, and sigmoidal. We balance the training sample setting the option `class_weight=‘balanced’`, which assign a larger weight to the HSPs. The results are shown in Fig. 5, with the rbf kernel reaching the best performance and optimum balanced-accuracy for C~8. We also test how the gamma parameter affects results, but the effect is negligible in our case; therefore, we set the ‘auto’ option.

**K-Neighbors (KN):** In KNN algorithm, the main optimisation parameter is the number of neighbours, which we scan from 5 to...
Table 4. Mean performance metrics for each ML algorithm, for the setup that maximises the balanced-accuracy. The means and $\sigma$ are calculated based on 300 re-samples. The latest column to the right ‘Opt-Acc’ corresponds to the optimum accuracy that could be reached with other setups.

| Classifier | Bal-Acc | Bal-F1 | HSP-Recall | HSP-Precision | LSP-Recall | LSP-Precision | Acc | Opt-Acc |
|------------|---------|--------|------------|---------------|------------|---------------|-----|---------|
| RF         | 0.926 ± 0.006 | 0.915 ± 0.006 | 0.959 ± 0.008 | 0.796 ± 0.015 | 0.892 ± 0.008 | 0.980 ± 0.007 | 0.913 ± 0.006 | 0.919 ± 0.006 |
| SVM        | 0.929 ± 0.005 | 0.915 ± 0.006 | 0.968 ± 0.008 | 0.795 ± 0.015 | 0.890 ± 0.009 | 0.984 ± 0.004 | 0.914 ± 0.006 | 0.914±0.006 |
| KNN        | 0.926 ± 0.006 | 0.910 ± 0.006 | 0.973 ± 0.007 | 0.780 ± 0.015 | 0.879 ± 0.009 | 0.986 ± 0.003 | 0.908 ± 0.006 | 0.909 ± 0.006 |
| GNB        | 0.914 ± 0.006 | 0.903 ± 0.006 | 0.946 ± 0.010 | 0.777 ± 0.015 | 0.880 ± 0.008 | 0.974 ± 0.005 | 0.897 ± 0.007 | 0.907 ± 0.006 |
| Ludwig     | 0.873 ± 0.015 | 0.887 ± 0.012 | 0.837 ± 0.029 | 0.801 ± 0.024 | 0.908 ± 0.012 | 0.926 ± 0.012 | 0.886 ± 0.011 | 0.886 ± 0.011 |

Figure 8. Balanced-Accuracy distribution for all the ML algorithms with optimal configuration, including results for the Ludwig auto-ML tool. The bottom density-box shows the distribution of individual bal-acc values.

For the GNB algorithm, we used Gaussian Naive Bayes (GNB): for the GNB algorithm, we
use a Gaussian Naive Bayes (GNB) model. The GNB model is a simple and effective
method for classification tasks. It assumes that the features are independent and normally
distributed with the same variance. The GNB model is less prone to overfitting than some
deeper models, but it can be less accurate in some cases.

5.3 Feature Importance

To explore the importance of each single-feature, one can try to visualise its impact on a given metric (e.g. balanced-accuracy) via a

Figure 9. Top: Correlation matrix showing Pearson’s coefficients to evaluate the linear correlation between features. Bottom: The four most correlated features, separated according to class: HSP (blue) and LSP (orange).
permutation test. However, this approach can only produce robust results in the case that all features are independent (or weakly correlated). If the data-frame has correlated features, an alternative is to apply the permutation jointly, to pairs of features, at once. Here we test both approaches and try to extract information related to the interpretation and consistency of our models.

The permutation test consists in splitting the data-frame into train and test, for training an ML model as usual. For the test sample, thou, the elements of each feature are shuffled or permuted. Consider ‘column $i$’ to represent a given feature $\alpha_i$; The shuffling is done separately for each $\alpha_i$, and each time, the ML model estimates the class of each object in the permuted-test-sample, is compared to the original balanced-accuracy of the test-sample (Strobl et al. 2007). In other words, if $\alpha_i$ is not correlated to other features, the information stored in each $\alpha_i$ is destroyed, and its direct impact on the balanced-accuracy metric is measured.

The permutation importance test does not depend on model parametrisation (e.g. to interpret model coefficients, Breiman 2001b), which renders it versatile to apply in all our cases. However, when features are correlated, e.g. features $\alpha_i$ and $\alpha_j$, the information about $\alpha_i$ is mapped into $\alpha_j$, and shuffling $\alpha_i$ alone is not enough to erase its information. As a result, the impact over the final balanced-accuracy is small and biased towards low importance.

To visualise the relation between features, we build the correlation-matrix, Fig. 9 (top), which shows the Pearson’s coefficient between all combinations of spectral slopes ($\alpha$). We do add an extra column (rdm) to our data-frame, consisting of random Gaussian and standardized noise ($\text{rdm}$) $\sim$ 0.0 and $\sigma$ $\sim$ 1.0. The random column works as a test-control, given we do expect negligible impact over the balanced-accuracy when shuffling it.

Indeed, our training data-frame has at least four pairs of spectral slopes with relevant correlation (Pearson’s coefficient $> 0.8$): $\alpha_{rw1}$, $\alpha_{w1w3}$, $\alpha_{w2w1}$, and a weak one for $\alpha_{rw1}$. We consider all those five pairs for a joint-feature permutation test. The scatter plots for the most relevant correlations are shown in Fig. 9 (bottom).

We carried out the single-feature permutation test with the python library mlxtend (MLX, Raschka 2018) mlxtend.evaluate. feature-importance-permutation. The MLX implementation has an internal option (num_rounds) which allows to randomly re-shuffle the feature permutation test n times (we adopt n=25) within a single loop, to probe the impact on balanced-accuracy. For the joint-feature permutation test we use the python library rfpimp which implements the shuffling of pairs of features ($\alpha_{rw1}$, $\alpha_{w1w3}$, $\alpha_{w2w1}$, $\alpha_{rw1}^{*}$, $\alpha_{w1w3}^{*}$, $\alpha_{w2w1}^{*}$). The results for both the single- and joint-features permutation tests are shown in Fig. 10.

To evaluate the degree of variability/uncertainty associated to each ML model, we run the entire test loop over 300 re-samples (for both the single and joint-feature permutation tests), and track the
mean value and standard deviations of the impacts on the balanced-accuracy metric. The models are trained at their optimum, as described in Sec. 5.2, and accounting for the unbalanced number of LSP & HSP sources (either via considering a class-weight correction, or applying SMOTE over the training sample). We use the random split option to produce the train & test samples (with 0.65 and 0.35 fraction, respectively) for each re-sample loop.

It is interesting to compare the results for the single- and joint-feature tests. For the joint-feature test, the most relevant pair of features is \( \alpha_{\nu_{\text{0}2}} \), for all models, followed by \( \alpha_{\nu_{\text{0}1.1}} \), \( \alpha_{\nu_{\text{0}2.3}} \) and \( \alpha_{\nu_{\text{0}1.3}} \). For the single-feature test, each model seems -misleadingly- to have a strong dependence in a particular spectral slope. The RF, SVM, KNN and GNB models have respectively: \( \alpha_{\nu_{\text{0}2}} \), \( \alpha_{\nu_{\text{0}1.1}} \), \( \alpha_{\nu_{\text{0}1.3}} \) and \( \alpha_{\nu_{0.3}} \) as the most important feature for each model, with no apparent reason behind the variability between models. However, the single-feature permutation test is biased because of the correlation between spectral slopes (as shown in Fig. 9), and the use of joint-features showed to be very useful to remove that bias.

The RF model had the largest \( \sigma \) for both single and joint-feature permutation tests. Note, this does not indicate the RF modelling is weaker (more unstable) concerning others, but that it is more sensible to the scrambling of a feature (or pair of features). In fact, the RF, SVM, KNN and GNB models are similar concerning model stability, as seen from the balanced-accuracy distribution in Fig. 8, given that the \( \sigma \) are similar (see Table 4).

The \( \Delta \) feature (introduced by hand) showed nearly no impact over the balanced-accuracy during the single and joint-permutation tests. This was envisaged as a test-control, to which we expected null permutation impact.

### 5.4 The intersection among classification models

In addition to the performance of each algorithm alone, we also compared the classification of each object among the models. The approach was to generate 1000 randomly split train and test samples, to guarantee that all objects appear in the test set a reasonably large \( n \) number of times (and smooth out fluctuations associated to \( n \)). We used these 1000 samples to train all the ML algorithms, including Ludwig.

Given the train and test process, we use two approaches to compare the classifications. The first one is to save the number of objects that are classified wrongly by at least one of the algorithms, in each of the 1000 test samples, what we call as missclassification. The result of this approach is that a mean of 148.4 objects are misclassified, i.e., there is a group with hundreds of objects accumulating around the miss-rate of \( \sim 350 \) counts, which follows approximately a Gaussian distribution having a mean of 348.7 ± 0.7 and \( \sigma \) of 17.1 ± 0.6. We interpret this group of objects as having an intrinsic ambiguity concerning the class they belong. Using a criteria of 5\( \sigma \), we assume a threshold at a miss-rate \( \geq 262 \) to mark (with a flag) these objects. Given the 1279 HSPs and the 2899 LSPs, 1073 (83.9\%) HSPs and 2499 (86.2\%) LSPs fall in the category with stable-classification; 206 HSPs and 400 LSPs got the ambiguity-flag. The flag will be useful for future works, to help refine the classification and better understand the ML-Blazar data-frame.

We investigate what could be the driver for misclassification, and found out that there is a growing number of ambiguous sources as the synchrotron peak approaches \( 10^{15} \) Hz, which is the threshold between HSP and LSP sources (border sources). Fig. 11 (bottom) shows a histogram according to \( \log(\nu_{\text{peak}}) \), separating between sources which are well classified most of times (miss-rate < 262) and cases which are relatively harder to classify (miss-rate ≥ 262). The histogram only considers HSPs because the \( \log(\nu_{\text{peak}}) \) information is only available for 3HSP sources (Chang et al. 2019). The 5BZcat has no \( \log(\nu_{\text{peak}}) \) information available. The fact that the ML models, as well as humans, tends to misclassify border sources indicates there would be no induced (or uncontrolled) cost by using ML instead of human classification, as discussed in Lipton (2016).

### 6 CONCLUSIONS

We build the ML-Blazar data-frame condensing multifrequency information on Radio, IR, Optical, and X-rays, for sources listed in the 5BZcat and 3HSP catalogues. The ML-Blazar sample includes 4178 objects labelled as 1279 HSPs and 2899 LSPs. Section 3 describes the cross-matching between catalogues and give details about the six flux channels available to describe each source.
The multifrequency fluxes were used to calculate spectral slopes ($\alpha$) that work as input features for training ML algorithms. In Sec. 3.2 we use statistical tests (KS and t-test) to compare the HSP and LSP distributions concerning each feature and select the five spectral slopes which better separate between HSPs and LSPs: $\alpha_{\nu_1}, \alpha_{\nu_2}, \alpha_{\nu_3}, \alpha_{\nu_4}, \alpha_{\nu_5}$.

In Sec. 4, we described five machine learning algorithms and the evaluation metrics used to train and test our blazar classification models. Those algorithms include Support Vector Machine (SVM), Random Forest (RF), K-Nearest Neighbours (KNN), Gaussian Naïve Bayes (GNB), and Ludwig (auto-ML framework). Besides, the training always corrects for unbalance via SMOTE-oversampling the HSPs, or by assigning larger weight to them.

In Sec. 5, we showed the influence of sample-size and test-fraction over the balanced-accuracy and shown that a 0.35 test-fraction can deliver low levels of uncertainty ($\sim 1\%$ $\sigma$ band) for the evaluation metrics, with no substantial penalty over the model’s predictive power. In Sec. 5.2, we describe the optimisation and best results for the SVM, RF, KNN, GNB, and Ludwig algorithms. We show that scanning over the parameter space available for each ML algorithm is an effective optimisation strategy. All data-points derived during the optimisation phase are mean values calculated from 300 re-sample, from which we extract the $\sigma$ as a measure of the uncertainty associated with the means.

We test the importance of the input features for the SVM, RF, KNN and GNB algorithms, via a joint-feature permutation test, that account for the correlation between spectral slopes. The pair of features that produce the main impacts in prediction power is $\alpha_{\nu_1},\alpha_{\nu_2}$, for all models, followed by $\alpha_{\nu_3},\alpha_{\nu_4}$ and $\alpha_{\nu_5}$. We have shown that supervised ML algorithms can be used for blazars classification into HSPs and LSPs based on multifrequency information, reaching up to $\sim 93\%$ of balanced-accuracy with the SVM algorithm. We had a focus in optimisation strategies to improve the balanced-accuracy for the SVM, RF, KNN and GNB algorithms, but also intend to provide the basis for future works.

Our perspective is to investigate neural networks applied to the classification task, and also for the autonomous determination of broadband spectral parameters (as the synchrotron $\nu$-peak). Provided the guide-lines, we envision the application of ML tools as a highly effective way to search for new blazars (and other types of sources) within current multifrequency databases, and prepare for the upcoming generation of deep-sky surveys.

ACKNOWLEDGEMENTS

BA is supported by São Paulo Research Foundation (FAPESP) with grant n. 2017/00517-4. PD is supported by São Paulo Research Foundation (FAPESP) with grant n. 19/08596-2. We made use of archival data and bibliographic information obtained from the NASA/IPAC Extragalactic Database (NED), data and software facilities from the SSDC and Opne Universe Portals managed by the Italian Space Agency (ASI) and collaborators. This research has made use of the VizieR catalogue access tool, CDS, Strasbourg, France (DOI: 10.26093/cds/vizier)

REFERENCES

Abdo A. A., et al., 2010, The Astrophysical Journal, 716, 30

Alam S., et al., 2015, ApJS, 219, 12

Amarantidis S., et al., 2019, MNRAS, 485, 2694

Arsioli B., Fraga B., Giommi P., Padovani P., Marrese P. M., 2015, Astronomy & Astrophysics, 579, A34

Arsioli B., Chang Y.-L., Musismenta B., 2020, Monthly Notices of the Royal Astronomical Society, 493, 2438A$2451$

Bai Y., Liu J., Wang S., Yang F., 2019, AJ, 157, 9

Ball N. M., Brunner R. J., 2010, International Journal of Modern Physics D, 19, 1049

Becciani U., et al., 2010, PASP, 122, 119

Boller T., Freyberg M. J., Truemper J., Haberl F., Voges W., Nandra K., 2016, Astronomy and Astrophysics, 103, 1

Borne K., 2009, arXiv e-prints, p. arXiv:0911.0505

Breiman L., 2001a, Statistical Science

Breiman L., 2001b, Statist. Sci., 16, 199

Breiman L., 2001c, Machine Learning, 45, 5

Brodersen K. H., Ong C. S., Stephan K. E., Buhmann J. M., 2010, in 2010 20th International Conference on Pattern Recognition. IEEE, doi:10.1109/icpr.2010.764, https://doi.org/10.1109/icpr.2010.764

Caruana R., Kangarloo H., Dionisio J., Sinha U., Johnson D., 1999, Proceedings / AMIA ... Annual Symposium. AMIA Symposium, pp 212–5

Chambers K. C., et al., 2016, ArXiv e-prints 1612.05560

Chang Y.-L., Arsioli B., Giommi P., Padovani P., 2017, Astronomy and Astrophysics, 598, A17

Chang Y. L., Arsioli B., Giommi P., Padovani P., Brandt C. H., 2019, A&A, 632, A77

Chawla N. V., Bowyer K. W., Hall L. O., Kegelmeyer W. P., 2002, Journal of Artificial Intelligence Research, 16, 321A$357$

Chiaro G., Salvetti D., Mura G. L., Giroletti M., Thompson D. J., Bastieri D., 2016, Monthly Notices of the Royal Astronomical Society, 462, 3180

Condon J. J., Cotton W. D., Greisen E. W., Yin Q. F., Perley R. A., Taylor G. B., Broderick J. J., 1998, The Astronomical Journal, 115, 1693

Cortes C., Vapnik V., 1995, Machine Learning, 20, 273

Cutri R. M., et al., 2013, Technical report, Explanatory Supplement to the AllWISE Data Release Products, http://adsabs.harvard.edu/abs/2013wise.rept....1C. http://adsabs.harvard.edu/abs/2013wise.rept....1C

D’Abrusco R., Massaro F., Ajello M., Grindlay J., Smith H., Tosti G., 2012, The Astrophysical Journal, 748, 68

Di Mauro M., Donato F., Lamanna G., Sanchez D. A., Serpico P. D., 2014, The Astrophysical Journal, 786, 129

Evans I. N., et al., 2010, Astrophysical Journal Supplement Series, 189, 37

Evans P. A., et al., 2014, The Astrophysical Journal Supplement Series, 210, 8

Gaia Collaboration et al., 2018, A&A, 616, A1

Ghisellini G., 1999, Astroparticle Physics, 11, 11

Giommi P., Padovani P., 2015, Monthly Notices of the Royal Astronomical Society, 450, 2404

Giommi P., Padovani P., Polenta G., Turriziani S., D’Elia V., Pranomonte S., 2012, Monthly Notices of the Royal Astronomical Society, 420, 2899

Giommi P., et al., 2018, arXiv e-prints, p. arXiv:1805.08505

Goldberger J., Rowes S., Hinton G., Salakhutdinov R., 2004, in Proceedings of the 17th International Conference on Neural Information Processing Systems. NIPSâZ04. MIT Press, Cambridge, MA, USA, p. 513A$520$

Helfand D. J., White R. L., Becker R. H., 2015, The Astrophysical Journal, 801, 26

Hodges J. L., 1958, Ark. Mat., 3, 469

Hopkins A. M., et al., 2015, Publications of the Astronomical Society of Australia, 32

Jones E. J., Oliphant T., Peterson P., et al., 2001, SciPy: Open source scientific tools for Python, http://www.scipy.org/

Kang S.-J., Fan J.-H., Mao W., Wu Q., Feng J., Yin Y., 2019, ApJ, 872, 189

Kelling S., Hochachka W. M., Fink D., Riedewald M., Caruana R., Ballard G., Hooker G., 2009, BioScience, 59, 613

MNRAS 000, 1–14 (2020)
This paper has been typeset from a TeX/LaTeX file prepared by the author.