Single-station meteor detection filtering using machine learning on MOROI data

Simon Anghel, Dan A. Nedelcu, Mirel Birlan, Ioana Boaca

Monthly Notices of the ROYAL ASTRONOMICAL SOCIETY

MNRAS **518,** 2810–2824 (2023) https://doi.org/10.1093/mnras/stac3229
Advance Access publication 2022 November 10

Single-station meteor detection filtering using machine learning on MOROI data

Simon Anghel [1,2,3]★ Dan A. Nedelcu,[1,2] Mirel Birlan[1,2] and Ioana Boaca[1]

[1]*Astronomical Institute of the Romanian Academy, 5 - Cutitul de Argint Street, 040557 Bucharest, Romania*
[2]*IMCCE, Observatoire de Paris, PSL Research University, CNRS UMR 8028, 77 Av. Denfert-Rochereau, F-75014 Paris Cedex, France*
[3]*Faculty of Physics, University of Bucharest, 405, Atomistilor Street, 077125 Magurele, Ilfov, Romania*

Accepted 2022 November 4. Received 2022 August 29; in original form 2022 June 16

ABSTRACT

Nowadays, extensive data are collected in an automated regime. Combining this, with the increase in accessible computational power, led to large-scale implementations of machine learning (ML). This is also the case of meteor science, where object detection often requires tracking of a moving light source between frames, and the number of false positives can be up to an order of magnitude higher than true meteoric phenomena. While spatiotemporal coincidence of events recorded by close, multiple cameras can eliminate most of the false positives, single-station detections in some camera networks are currently discarded. In this paper, we explore a set of ML models aiming to find an optimal method for re-analysis of this single-station observations, in order to identify and extract real meteors. A set of 15 ML models were trained on features extracted from the meteor movement. Upon testing, we found a top accuracy score of 98,2 per cent, and a recall (i.e. percentage of meteors correctly classified) score of 96 per cent for the best performing models. When combined with the spatiotemporal coincidence of the detection, the recall increases to 99.92 per cent. These 15 ML techniques were selected according to their ability classify tabular data, hence the bundle can be applied to other studies. The same goes for the computed features, which are independent on the camera configuration, thus, the process can be scaled and applied to other networks. These methods are to be implemented to re-analyze the events recorded by the larger, FRIPON network.

Key words: methods: numerical – techniques: image processing – meteorites, meteors, meteoroids.

1 INTRODUCTION

Obtaining a variable of interest from multiple dimension data set usually involves plenty of laborious checks and mostly dull tasks such as filtering, discrimination, and comparisons between several attributes. This is where machine learning (ML) intervenes to filter out the unusable structures, and accurately extract the information deemed relevant. The first algorithms of this kind were quite rudimentary and involved supervision. This process was first introduced by Rumelhart, Hinton & Williams (1986), and reinvigorated for large networks by Hinton & Salakhutdinov (2006). These ML routines have experienced an accelerated development lately, and became widely used across all science disciplines, as their non-linear fitting methods were proven very efficient when working with big data (e.g. Krizhevsky, Sutskever & Hinton 2012, and the references therein).

Within an optical fireball network, the detection procedure involves the tracking of moving sources of light across the detector. The detection of such phenomena implies either grabbing long exposures continuously and search the images for meteors, or analysing the live video obtained from the camera in real time, and store only the segments which involve a moving light source. When targeting single-station detections, both methods require a manual validation of the user by either finding the objects in the images, or filtering the false positives (FP) from the true meteors. Upon this procedure, there are cases on the border between classes, and this may result in adding false meteors in the true category due to insufficient information. Also, in many situations the action of validation becomes tiring and prone to mistakes, and will introduce user bias in event validation. Hence, by implementing an automated ML detection filter will unburden the user, allowing time for other scientific work.

Depending on the detection procedure, different methods can be used to tackle this problem with the aid of ML. In the case of meteor detection using the long exposure captures, an image recognition model can be integrated in the pipeline to detect streaks of light, as illustrated by Galindo & Lorena (2018). The authors applied a pre-trained Convolutional Neural Network (CNN) to detect meteor from images, obtaining a 96 per cent accuracy. A completely different approach in meteor detection filtering was employed by Silađi, Vida & Nyarko (2015). The authors derived several features from the meteor movement across the image. Thus, by employing soft computing algorithms, they obtained a binary classification accuracy score of ≈90 per cent. The score could have been improved with a larger data set, however, the authors aimed to test the validity of the algorithms. This is the closest approach in meteor classification from the algorithms discussed in this study. In a separate study, De Cicco

★ E-mail: Simon.Anghel@astro.ro

et al. (2018) tested a random forest, a CNN and a Long Short Term Memory (LSTM) for their ability to precisely classify detections in CAMS data. The results vary from 88–90 per cent precision (i.e. the percentage of predicted meteors to be real meteors) on a severely imbalanced data set, being comprised of 200 000 detections, of which, only 3 per cent are meteors. On a same size data set of CAMS (this time, a balanced one), Gural (2019) employed several deep learning algorithms to filter meteor detections. Thus, obtaining a recall scores (properly classified true meteors) ranging from 98.1–99.94 per cent, surpassing those of an experienced human user. Recently, Towner et al. (2020) compared two methods for fireball detection filtering, with results ranging from 67–82 per cent for small fireballs. By introducing the network geometry considerations (for double- to multistation detections), the scores increased to 99.7–99.9 per cent.

Currently, within the Fireball Recovery and InterPlanetary Observation Network (FRIPON) consortium (Colas et al. 2020), the detection procedure is conducted automatically, via the FREETURE software (Audureau et al. 2014), which is prone to false alarms. Meteor detection is achieved by locating zones on the video frame which exceed an arbitrary threshold. The zones passing this filter are tracked between frames. If the locations follow a quasi-continuous line in successive frames, the detection is considered to be a meteor. This moving object localization procedure is also prone to false detections, as there can be multiple sources of light which move across the frame with angular velocities resembling those of meteors. The number of FP could be decreased by only keeping detections with high angular velocities (higher than several degrees per second). On the one hand, this filter would drastically reduce the number of satellites or aeroplanes. On the other hand, it would also increase the risk of missing slow-moving meteorite-dropping fireballs. These have a high scientific value, as the data can be used to derive the pre-atmospheric properties of the object (e.g Anghel et al. 2021(c)), and constrain the entry models. Therefore, the detection is tuned to also consider slow objects, leaving the user with a high number of false positive to manually inspect (≈90 per cent of the detections).

Some of the true events can be obtained automatically with a coincidence filter, by comparing the data from two or more stations for the same event. However, this process will leave a large number of single-station fireballs unanalysed. By including these objects will help to enhance our understanding of cm-sized meteoroid flux. One of the aims of the FRIPON network is to better constrain this flux (Colas et al. 2020), a currently understudied area of meteor astronomy, since the work of Halliday, Griffin & Blackwell (1996). Other groups are also working on this problem (e.g. Vida et al. 2022), since the flux density is important to assess the risk for the civilian infrastructure in space. The aim of this study is to train a set of soft computing algorithms (i.e. ML) on a set of parameters extracted from the detection (e.g. meteor duration, direction, positional measurements), to automatically distinguish between meteors and non-meteors (e.g. satellites, aeroplanes, reflections). The resulting single-station true detection data set contains scientific data that can yield useful photometric information (i.e. details of fragmentation and light curve), and approximate position of the radiant. Moreover, these methods could be also applied to classify satellites, extending the application to Space Surveillance and Tracking (SST) activities. The advent of Low Earth Orbit satellites mega-constellations will severely interfere with the meteor detection network operations requiring automated methods for identification, characterization, and classification of transient events.

2 DATA PREPARATION

2.1 The network

All sky camera networks focus on meteoroids towards the larger end of the mass scale (cm sizes and larger). Their optical setup usually results in a lower resolution, less sensitive detections. The main operational goal of such networks is meteorite recovery and linking the found meteorites with a dynamical context (e.g. Weryk et al. 2008; Cooke & Moser 2012; Bannister, Boucheron & Voelz 2013; Howie et al. 2017; Peña-Asensio et al. 2021). The Meteorites Orbits Reconstruction by Optical Imaging (MOROI) network (Nedelcu et al. 2018) is operated by the Astronomical Institute of the Romanian Academy (AIRA), and it is part of the FRIPON consortium (Colas et al. 2020; Gardiol et al. 2021). During 2017–2020 the network extended to 11 cameras (Anghel et al. 2019), sending each detection to the central server in Bucharest, to perform the trajectory and orbit computations (Fig. 1). The MOROI cameras are based on the 4.9 × 3.6 mm Sony CCD chip ICX445 (1296 × 966 active pixels, 12 bit dynamic range), resulting in a 3.75 × 3.75 µm pixel size. In fast frame rate mode (30 FPS) the camera provides the temporal resolution necessary to measure fireball velocity and deceleration along the trajectory.

Each MOROI station uses FREETURE software (Audureau et al. 2014) for both meteor detection and image acquisition. The software is fully automated and can be adjusted for nighttime or daytime observations. In order to calibrate the events and to characterize the sky quality, the software collects a 5-s (20 μs during daytime) long exposure image (i.e. *capture*) every 10 min, which is stored in `FITS` format. In the image header we can find the camera settings along with spatiotemporal information necessary for the star matching sequence (Anghel et al. 2019).

The meteor detection algorithm is optimized to reduce the cost of image processing (Audureau et al. 2014), thus, the images are downsized by a factor of two, and positions are returned as integers. This will in turn trade accuracy for speed (i.e. to capture as many positions as possible in the shortest time). Hence, the meteor positions will be truncated (Fig. 2), and unreliable for trajectory computation. Hence, a pro per centroid extraction is applied before computing the trajectories, to ensure accurate positions, for reliable orbit computations (Anghel et al. 2021(a)).

As of 2020 September, we started the integration of each new MOROI camera with the FRIPON consortium (Anghel et al. 2021(b)), and currently there are 16 operating cameras in Romania, most of them already sending each detection and calibration data to the FRIPON data base, located in Marseille (France). The consortium delivers an interface to visualize the recorded data and the processed results via an open-data base.[1]

For this study, we focused on the events recorded within the MOROI network between 2017–2020. Each event consists of a data package of roughly 1 Gb, containing the frames of the detection (in `FITS` format), video, and composite frames. In addition, the event data also contains a file (`positions.txt`), which records the approximate *x, y* image approximate coordinates of the meteor detection, the time stamp, and frame number, as obtained in the detection thread (Audureau et al. 2014). This positional data was used to compute the features for model training.

[1] https://fireball.fripon.org/

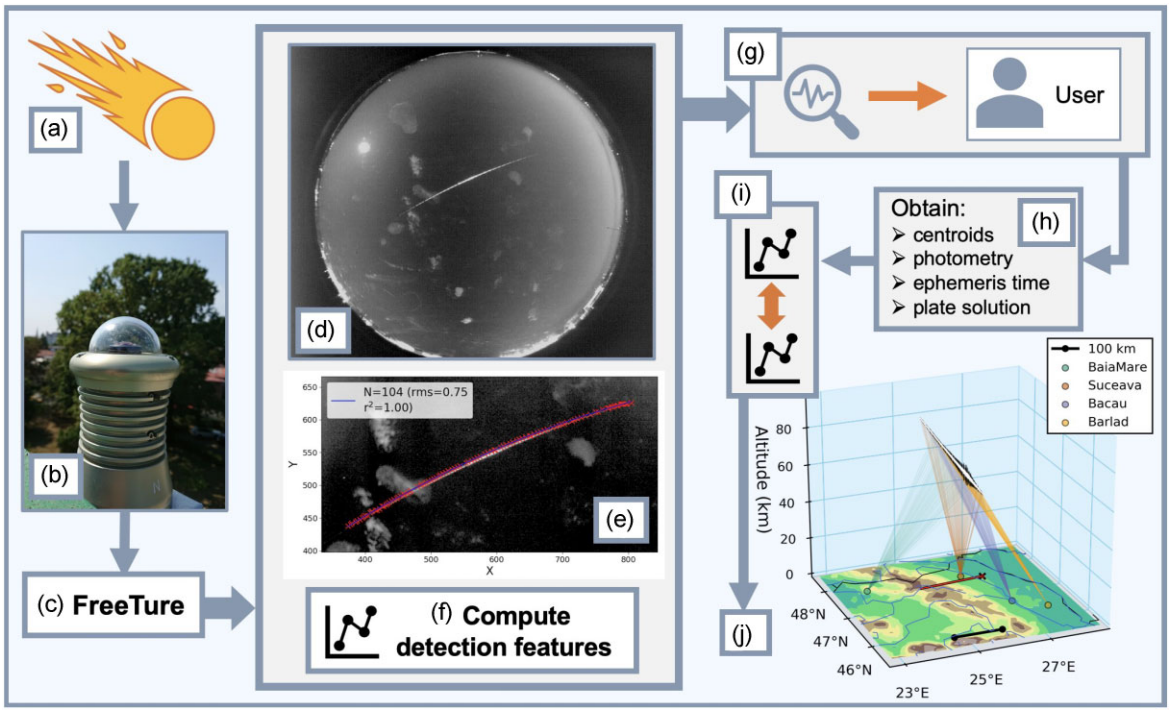

Figure 1. Schematic of the MOROI network data flow. (a) The luminous phenomenon can either be a fireball or a false event (e.g. city lights, reflections, aeroplanes, satellites). (b) An all-sky camera which monitors the sky (in here is the camera mounted at the AIRA). (c) The FREETURE software runs a background detection loop on the acquisition and stores each event (Audureau et al. 2014). (d) The stacked frames of the event. (e) The image positions of the detection superimposed on a cutout of the stacked frames. (f) A set of features (e.g. radial velocity, duration, direction) is computed from the `positions.txt` file. (g) The filtering of true and false detections. This can be employed by the user or a trained ML model. (h) Compute accurate meteoroid positions, obtain the photometry and celestial coordinates. (i) Merge the data from this station with data from other stations of the same event, which is also filtered by the previous steps. (j) Reconstruct the atmospheric trajectory of the meteoroid.

2.2 Detection data

For this study, we gathered all the events (totalling 9858) detected in MOROI data base on the course of 4 yr (2017–2020). Some events displayed a clear non-meteor behaviour (i.e. 1–2 recorded positions). These are associated with hot pixels, comic rays, etc., and were discarded from analysis. Only the events with 3 or more frames were taken. The rest (N = 8086) are kept as the *working set*. During the supervised validation procedure (see Section 2.3), this was found to be comprised of 6922 false detections and 1164 true meteors. Among the true meteors, only ≈ 20 per cent are multistation detections. The data was obtained from 10 all-sky cameras, thus representing a wide variety of sky-backgrounds level, horizon mask profiles, and general sky conditions.

2.2.1 Feature extraction

In the feature computation process, first the meteor region of interest (ROI) is prepared with a best fit routine on the `positions.txt` file. Currently, the FREETURE data reduction procedure is optimized for speed, at the cost of accuracy. Because of this, from time to time there are positions missing from the detection frames (Fig. 2) and will not allow a safe transition towards the matching stage. With this in mind, a cleaning and refitting routine is applied iteratively, on one point at a time, to a total of 30 per cent of the data points in the detection. The cleaning procedure is first applied to the *x vs frame* polynomial fit, followed by the *y vs frame*. Next, an interpolation procedure is applied to obtain the objects estimated

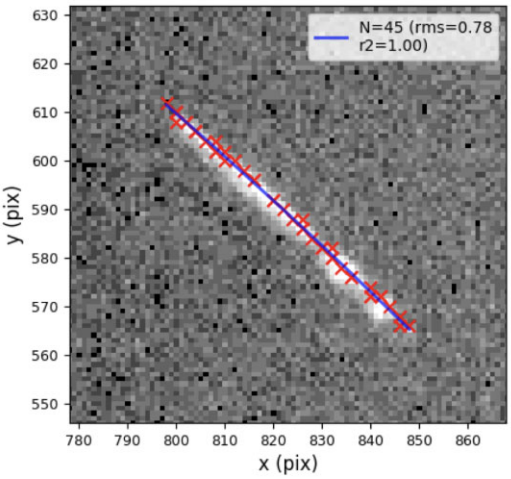

Figure 2. The meteor positions superimposed on the event region of interest. The meteor position (*x, y*) generated by FREETURE is marked with a red cross for each frame (168 to 224). The blue line is a second degree regression. The *rms* (measured in pixels) and *r2* are the root mean square and the coefficient of determination respectively, for the rough meteor positions.

positions in the missing frames. The resulting line, represents the best fit prediction for the coordinates of sources in the same frame range. This procedure is applied to each event, which are then used to compute positional features of the object's motion on the image. The entire list of features computed for model training includes:

(i) **range_centre** - distance between the average event position and the image centre.
(ii) **sum_x** - the sum of x coordinates of the detection.
(iii) **sum_y** - the sum of y coordinates of the detection.
(iv) **sum_frame** - the sum of the frame numbers from the detection
(v) **avg_x** - the average position of the event measured on *x*-axis.
(vi) **avg_y** - the average position of the event measured on *y*-axis.
(vii) **min_x** - the minimum position of the event measured on *x*-axis.
(viii) **min_y** - the minimum position of the event measured on *y*-axis.
(ix) **min_frame** - the minimum frame number.
(x) **n_frames** - the number of frames involved in the detection.
(xi) **duration** - the meteor duration measured in seconds between the first and last frame.
(xii) **frame_span** - the number of frames in the event, obtained by subtracting the first frame number from the last.
(xiii) **predict_span** - the number of predicted frames obtained through interpolation. On true events, this number is close to the frame span.
(xiv) **pix_speed** - the apparent speed of the meteor (pixels/frame).
(xv) **xcomponent** - the x component of the pix_speed, measured in pixels.
(xvi) **ycomponent** - the y component of the pix_speed, measured in pixels.

The last set of feature computations include several methods of regression analysis on the meteor CCD positions (Fig. 2):

(i) **rmse_deg1** - the root mean square error (RMSE) from the simple least squares line, computed for (*x*, *y*) positions.
(ii) **r2_deg1** - the coefficient of determination for the least squares line, computed for (*x*, *y*) positions.
(iii) **rmse_deg2** - the RMSE from the least squares line of the second degree, computed for (*x*, *y*) positions.
(iv) **r2_deg2** - the coefficient of determination for the second degree regression on meteor positions.
(v) **rmse_t_deg1** - the RMSE from the simple least squares line, computed for the transposed (*x*, *y*) positions.
(vi) **r2_t_deg1** - the coefficient of determination for the simple least squares line for the transposed (*x*, *y*) positions.
(vii) **rmse_t_deg2** - the RMSE from the least squares line of the second degree, computed for the transposed (*x*, *y*) positions.
(viii) **r2_t_deg2** - the coefficient of determination for least squares line of the second degree, computed for the transposed (*x*, *y*) positions.

The above list of features ($N_{features} = 24$) along with of the location-specific data (i.e. station name, location, date, etc) are stored in a table to be operated by the ML algorithms. A subset of the features, are also used to generate a detailed user validation window, to display the important aspects of the event data. To identify dependencies and spot key components between features in the list, these were displayed in pair plots and visualized (e.g. Fig. 3).

Most of the features involved in false events tend to have some common aspects, such as likelihood to be triggered close to the horizon. When plotting the **avg_x** vs **avg_y** for each event in the set, a pattern emerges (Fig. 4). Near the horizon there are separate contours of false events gathered in clusters. These are common station-specific positions on the horizon where cars turn, and the headlight behaviour will trigger a meteor detection. Though these FP can be filtered with a mask around the horizon, for this study we kept these events, as they tend to mimic meteors well, enriching the examples used in model training.

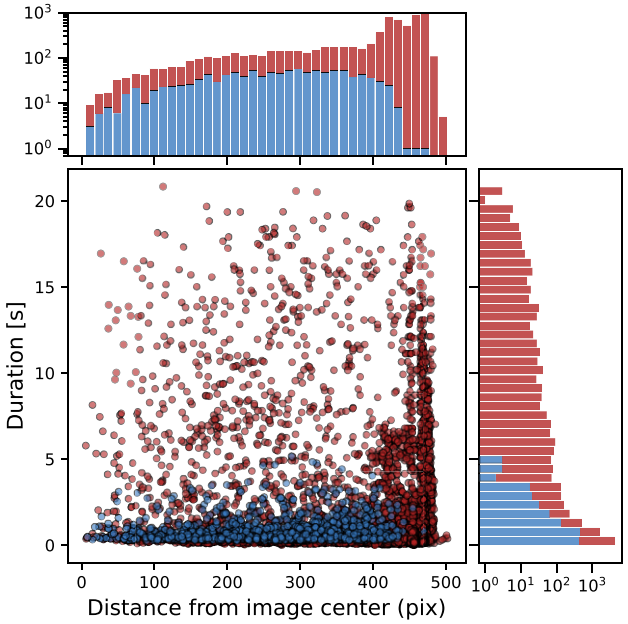

Figure 3. Example of feature dependence visualization displaying the event **duration** as a function of distance from the centre of the image (**range_centre**). The data consists of 6922 false (red) and 1164 true (blue) events. The histograms for both axes are displayed on a logarithmic scale.

2.2.2 *Feature selection*

To keep only the relevant features, a *keep-best* algorithm was implemented. This cycled through all the computed features, adding one feature at a time, and testing the model score; if the score improved (by a set margin), the feature is kept in the list, and the algorithm moved to the next feature, and repeats the test. If the score does not improve, the feature is skipped, and then the procedure moves to the next feature. This was performed multiple times using randomized order of the feature list to obtain a final feature list for model training.

2.2.3 *Correlation between features*

Having multicollinear features, would often cause the models to return nonsensical results and unreliable estimates. Thus, to spot the dependence between features, a correlation matrix was obtained. For two random variables this can be defined as:

$$\rho_{X,Y} = \frac{cov(X, Y)}{\sigma_X \sigma_Y}, \tag{1}$$

where ρ is the correlation coefficient, *cov* is the covariance, and σ_X, σ_Y are the standard deviations of the X and Y variables. For two sets of features X_i and Xj, where $i, j = 1, 2,..., n$, the correlation is defined as

$$cor(X_i, Xj) = \frac{cov(X_i, X_j)}{\sqrt{V_{ii} V_{jj}}}, \tag{2}$$

where V_{ii} are elements in the covariance matrix. The coefficient can take values from $-1.0 \leq \rho_{X_i X_j} \leq +1.0$ as described by Weisstein (2022). A representation of the cross-correlation between the computed features is displayed in Fig. 5. Obtaining the correlation score

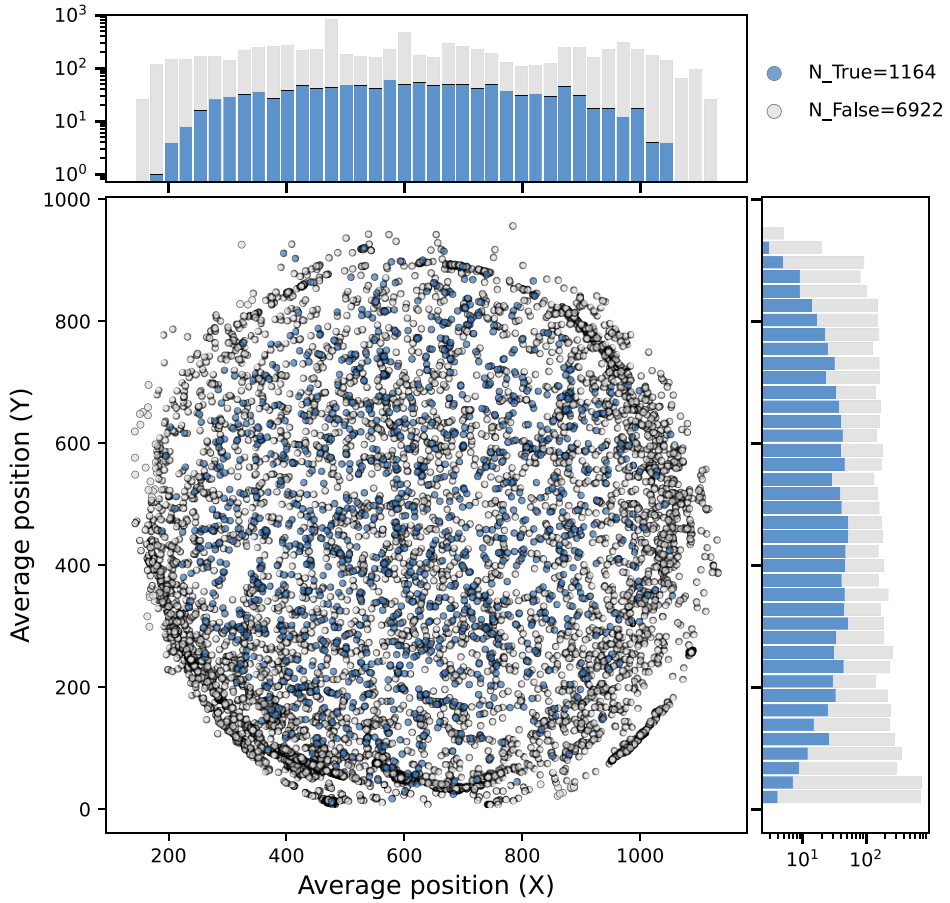

Figure 4. The average position (x, y) of the set of events from all stations (in 2017–2018). On top and right side of the image, a histogram with the total number of detections in log scale is shown. Note the concentration of false events close to the horizon line due to cars/city lights.

is a starting point for the investigation, as this allowed us to better understand each component of the data set and select the relevant features for the model training. Regarding our data set, the majority of the features show a poor correlation score, which is ideal when training the ML models.

2.3 Supervised validation of MOROI data set

To label the data into either meteors or non-meteors, the *working set* (N = 8,086) was processed by extracting the positions involved in the detection, applying the procedures detailed in Section 2.2, and cropping the associated ROI from the composite image of the event.

However, from the ROI alone, it is rather difficult to distinguish meteors from satellites/planes. Thus, to better perform the supervised validation, a set of complementary features of the event were also displayed along the meteor image, as presented in Fig. 6. These include several features presented in Section 2.2 (i.e. **n_frames**, **rmse_deg2**, and **r2_deg2**). In addition, the window also includes information of the direction of motion, as there might be events which appear to have a certain orientation in the image, while the data indicates otherwise. Finally, the detection position is displayed in the full all-sky image (Fig. 6).

As previously stated, the meteor detection algorithm is operated with liberal detection parameters which ensure that meteors are not missed, but these settings cause a large number of FP. For example, Feleac station triggered a large ratio of false detections by either aeroplanes approaching or departing the nearby airport, or from car headlights on a road curve towards the northern part of the image (Fig. 6). Another major false detection area was in the vicinity of the building in the northern part of Berthelot station during 2017–2018 before relocation. Currently, the Berthelot station displays the darkest sky among the stations in the network (Anghel et al. 2019; Birlan et al. 2021).

Data visualization also helps to track the behaviour of the light curve. Aeroplanes are usually equipped with an intermittent signal light, and depending on the type of aeroplane, and the position on the sky, the trail of light marks the image with different patterns. While performing the manual classification, we noticed the following:

(i) if the light curve is flat, this suggests it is a satellite.
(ii) aeroplanes close to the horizon do not usually display the intermittent light. These however, sometimes reflect the sunlight during twilight and resemble meteors the most.
(iii) when analysing pixel speed, aeroplanes, and satellites usually travel with an apparent velocity much slower than the average meteor.

For those events which sit at the borderline between meteor and non-meteor behaviour, the images must be inspected using additional filters. A mosaic of different examples of event ROIs is displayed in Fig. 7. Moreover, there are few scenarios when complementary data (e.g. duration, meteor speed) needs to be analysed before classifying the event as true or false.

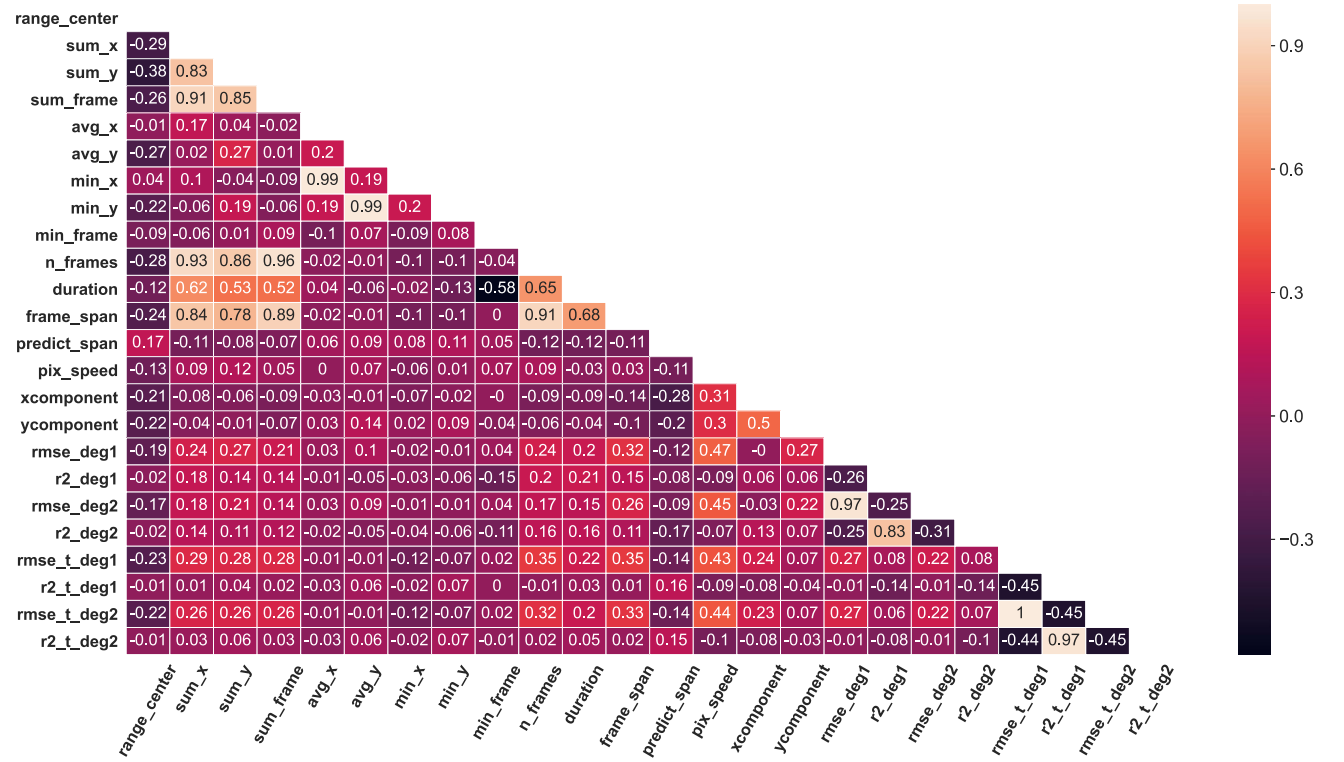

Figure 5. The correlation matrix of all the features computed for meteor classification, defined in Section 2.2. Note the neutral aspect (with some exceptions) of the correlation between the features.

3 METHODOLOGY

Most existing meteor networks validate detections as real meteors by performing a correlation in time and space by confirming that the event was observed at the same time by several different cameras at the expected range of meteor heights (e.g. Jenniskens et al. 2011; Vida et al. 2016). While this is the method used also for trajectory computations, it requires at least two stations to spot the meteor.

On the average network, the number of single-station detections far exceeds the number double- to multistation events. This is caused mostly by false detections, followed by meteors observed at the edge of a network or when clouds impede the meteor light, and rarely when stations are being switched on to calibration image acquisition mode. In the case of MOROI network during 2017–2020 period, there have been five times more of real single-station meteor detections than validated multistation detections. Thus, to automatically identify single-station meteor events, we test which ML models performs best on this task.

3.1 ML models

The models can vary a lot between the ways data are selecting or dividing the group, the boundary selection, order of the steps, etc. The general idea is to use the software to find patterns in the data, and extract only the needed parameters (defined by the user), or the one proposed by the software to be distinctive. For the ML experiment we implemented the models via *scikit learn*[2] library (Pedregosa et al. 2012) using PYTHON open source programming language. The model selection started from a full implementation of the models suited for classification. After several tests with the data, only the models found relevant for binary classification (i.e. able to classify the data into two categories, meteors or non-meteors), which returned realistic results, were kept. This resulted in a set of 15 ML algorithms. The list includes:

(i) *Nearest Neighbors* - searches for the closest distance between training samples and the new point (Altman 1992).

(ii) *Decision Tree* - a set of rules based on thresholds which increase in complexity until all data are classified (Breiman et al. 1984).

(iii) *Extra Tree* - an extremely randomized tree classifier (Geurts, Ernst & Wehenkel 2006).

(iv) *Adaboost* - a combination of a set of weak learners, on repeatedly modified versions of data (Freund & Schapire 1997).

(v) *Gradient Boost* - optimizes loss functions using an additive model in a forward stage-wise fashion (Friedman 2001).

(vi) *Random Forest* - averaging algorithms based on randomized decision trees (Breiman et al. 1984).

(vii) *Stacking Estimators* - Stacks the output of individual estimators and use a classifier to compute the final prediction (Wolpert 1992).

(viii) *Neural Net* - a classic multilayer perceptron classifier (Hinton 1990).

(ix) *Gaussian NB* - Naive Bayes theorem approach for supervised learning algorithm (Rennie et al. 2003).

(x) *Linear SVM* - a Support Vector Machine based linear classifier using the largest separation hyperplane (Chang & Lin 2011).

(xi) *RBF SVM* - radial basis function used as a kernel in Support Vector Machine (Chang & Lin 2011).

(xii) *QDA* - dimensionality reduction method using Bayes rule (Hastie 2008).

[2]https://scikit-learn.org/stable/

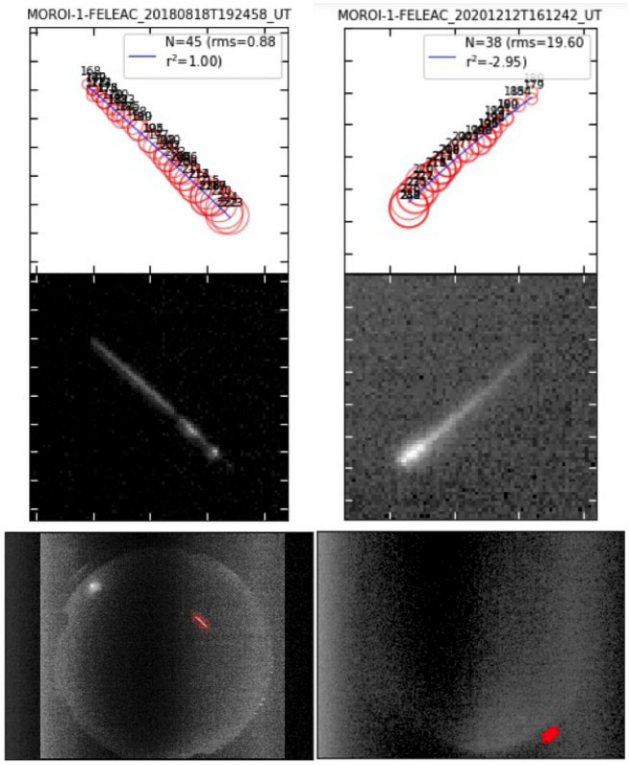

Figure 6. Two examples of events which triggered a detection (left: meteor, right: non-meteor). Both events are displayed as a column plot. This is comprised of (starting from top panel) a representation of the event data (x, y) reduced by FREETURE, along with the number of frames, the root mean square error (*rms*) and coefficient of determination (r^2) computations. The increasing size of the positions (circle) marks the direction of motion. The middle row displays a ROI of the event composite image, and the bottom row displays the full all-sky image with the location of the event marked. The left column constitutes a true meteor with double fragmentation. The right column is a car headlight at the horizon, signalled by the detection procedure.

(xiii) *Perceptron* - A linear perceptron classifier suitable for large-scale learning. It updates the model only on mistakes (Freund & Schapire 1999).

(xiv) *Ridge* - a regression model where the loss function is the linear least squares function (Rifkin & Lippert 2007).

(xv) *Stochastic GD* - regularized linear models implemented using the stochastic gradient descent learning (LeCun et al. 1998).

3.2 Training & evaluation

In the supervised learning process, the user must construct a model input from a set of independent variables and manually classify (label) each data point. During training, the ML algorithm finds the set of model coefficients which reproduce best the manually labelled data set.

3.2.1 Model training

One of the objectives of this study was to find the model which best predicts the data and optimize its binary classification score. A model, which is tested on the same data it trained, would learn the particularities of the data set and will begin to overfit, returning a 100 per cent accuracy. However, we would not know how well it behaves on newly seen data (i.e. new detections). Thus, during model training, a set of unseen data was kept for testing the model accuracy.

On the one hand, keeping a high train-to-test ratio will almost ensure a high score. This is mostly due to a low test count, thus, a lower chance of erroneous classification. On the other hand, a lower train-to-test ratio, will lead to a poorly constrained model, and accuracy will decrease. For this study, the train-to-test splitting ratio was set to 70:30, resulting in 5660 for training and 2426 data points for testing. This splitting procedure was applied 10 times using a randomized version of the *working set*, and the scores were averaged to obtain a final result.

Upon a brief inspection of the events, some features were found to be important when distinguishing between meteors and false alarms. These were computed beforehand, and examined amid data labelling (Fig. 6). One important feature is the average distance from the image centre, the other being the duration (in seconds) of the event (Fig. 3).

As an exercise in model behaviour, all 15 ML models were tested with every two feature combination from the 24 features extracted from the data FREETURE data (Section 2.2). A model decision boundary representation can be found in Fig. 8, for two features found to be very significant – the event duration and the distance from the image centre. The purpose of this operation is to visualize the model classification techniques, characteristic only for two-feature separable data sets. When we increase the number of dimensions (features), the boundaries shift into hyper-planes, which become increasingly harder to represent.

3.2.2 Hyper-parameter tuning

Most ML models have different parameters to tune in order to perform optimally for the nature of input data. A perfect fit of the training set will result in overfitting, and will be detrimental when the model is presented with never-before-seen data. Thus, a model regularization procedure is needed to achieve a high classification accuracy without overfitting. The process is done by training the model using different combinations of hyper-parameters, and testing it on the evaluation data set (unseen by the fitting procedure). The hyper-parameters control either aspects regarding the weight of each feature or how the learning coefficients are assigned.

Let us take as example the *Nearest Neighbors* function. When searching the best hyper-parameters, one could examine the possible algorithms of computing the distance (metric) between each neighbour and the new data point, the number of neighbours considered by the model and the weight function. The latter can be also added as a list of values by the user. Depending on the hyper-parameters used, certain aspects can be learned about the model behaviour by plotting the scores during the tuning phase. Fig. 9 displays a clear increase in classification accuracy when using the *canberra* distance d, as opposed to other commonly used distance functions. The *canberra* distance is given by Emran & Ye (2002) as follows :

$$d(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{n} \frac{|x_i - y_i|}{|x_i| + |y_i|}, \tag{3}$$

where $\boldsymbol{x} = (x_1, x_2, ..., x_n)$ and $\boldsymbol{y} = (y_1, y_2, ..., y_n)$ are n-dimensional vectors representing the feature values, when comparing several metrics. Also, when increasing the number of neighbours over $\approx$20, the model is generalizing across too many data points (i.e. it 'steals' some of the data from the other class) and begins to perform poorly.

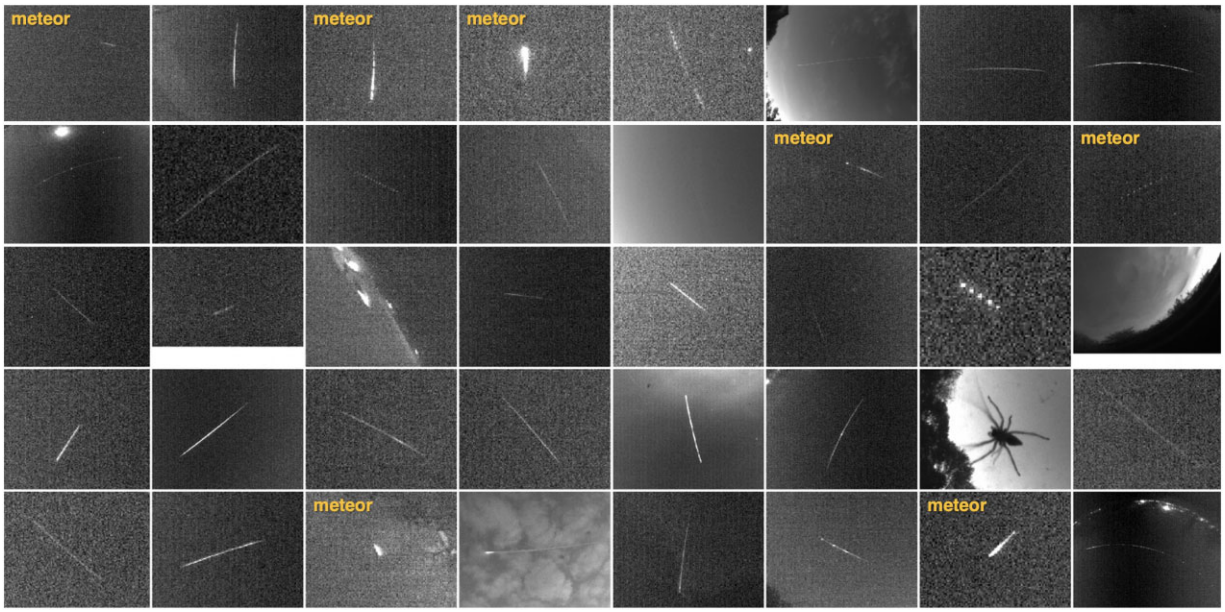

Figure 7. A mosaic of detections. Each image represents a zoom-in of a part, or full location on the image where the event occurs, used in the inspection phase of supervised validation. Note that most of the images are false detections, representing the path of a plane or satellite. The spider is an example of false detection due to moonlight reflexions on one of its moving legs.

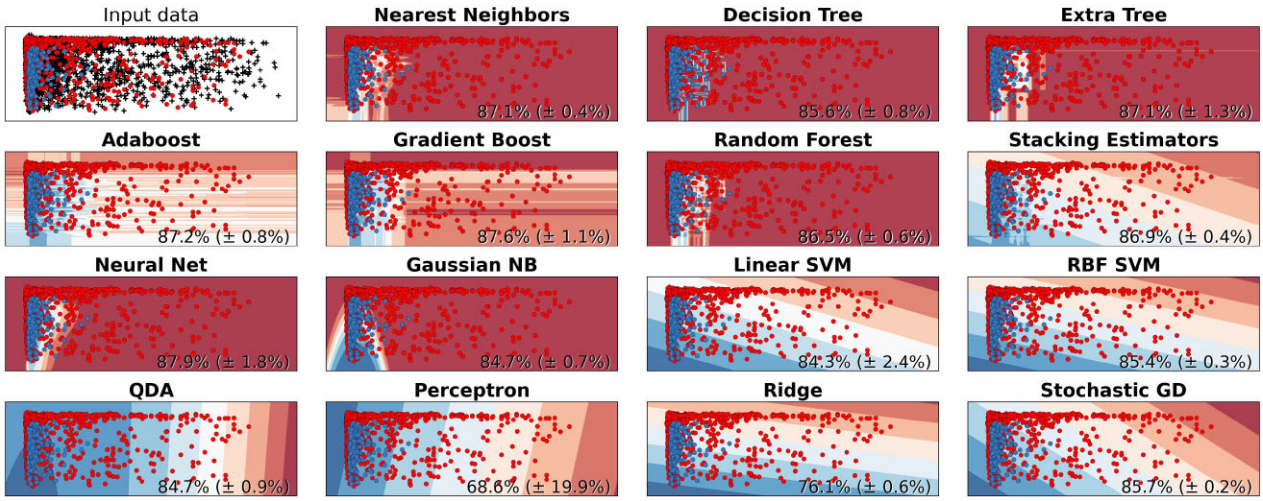

Figure 8. A comparison between the nature of decision boundaries of the models used in this analysis. The representation is obtained via a classification based on two features (i.e. meteor duration on x, and distance from image centre on y). This was obtained first by normalizing the data (i.e. the features were flattened to values ranging from −1 to 1), followed by a split into train and test. The train set (i.e. black crosses in the Input data) was used to obtain the model-specific decision boundaries. These are represented by the background heat maps, where the zones populated by false and true meteors are distinguished by a gradient of red and blue colours respectively. Finally, the test data (coloured circles) are superimposed on the decision boundaries, divided into False (red) and True (blue).

To find the best hyper-parameters, each model was tested with the final set of features (see Section 2.2) using all combinations from a prepared set of values. This was employed using the `GridSearchCV()` function in *scikit-learn*. This function builds a grid based on all the individual values selected by the user and tests the model using one hyper-parameter combination at a time. Taking the example of the *Nearest Neighbors*, the optimal tuning for the meteor classification problem was by using a *canberra* metric on 18 nearest neighbours in manifold feature space, based on an 'uniform' weighing system. The full set of tested hyper-parameters for the models used in this study, along with the best values are presented in Table B1.

4 RESULTS AND DISCUSSION

Ten sets of events from the MOROI data base (i.e. one from each station) were processed to experiment with fifteen ML models for filtering meteors. After the preparation and supervised validation, the data were randomized for a total number of 10 times. For each randomization, the working set was separated into 5660 events for training, and a test group of 2426 (ratio of 7:3), and used as input for the models. After the *keep-best* feature selection cycle (see Section 2.2.2), 7 features were found to increase the score, among the list of 24 computed. A pair plot of the final feature list is displayed in Fig. 10. This combination of features resulted in a score improvement

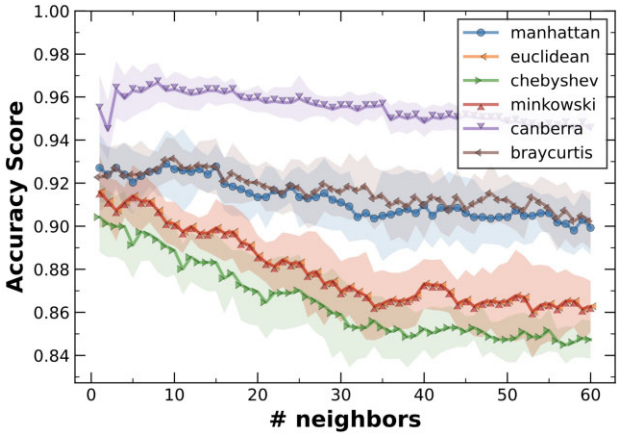

Figure 9. Hyper-parameter tuning applied to *Nearest Neighbors* function. The graph is displaying the classification accuracy as a function of the number of neighbours for a set of different metrics. The shaded area corresponds to the standard deviation of the score extracted from 5 cross validations, employed on the final data set.

of $\approx$ 7 per cent compared to the two most representative features (Fig. 8).

4.1 Scoring metrics

In the field of ML, the performance of the model can be visualized with a specific table configuration, called a confusion matrix, or error matrix (Stehman 1997). Rows in this table represent manually labelled data, and columns represent the labels predicted by the algorithm. This layout allows us to visualize whether the model mislabels the results (i.e. confuses the predictions with respect to the actual targets). This is represented as a table with equal number of rows and columns (i.e. for the actual targets and the predicted), each corresponding to a class in the data set. For this study, the confusion matrix represents two classes, and in the matrices, the results are displayed as a ratio. With the models tuned, and the top 7 features obtained, a sequence of 10 randomized train and test runs is conducted to obtain the final classification scores (Fig. 11). Although the result can be represented as label counts (i.e. the number of instances for each square), in case of a group imbalance (we have 6$\times$ more FP than real meteors), a normalized translation is preferred. For this, we use the broad terminology adapted for our each group in our study (Fig. 11).

To evaluate the performance of a model, an additional set of metrics were computed by using the confusion matrix results as input. The full list of metrics used in this analysis, and examples are detailed in Appendix A. We used the *accuracy* to measure how close the prediction is from the given label (the higher, the better); *precision* which focuses on the ratio of the correctly predicted meteors; *recall* to obtain the TP rate (higher is better), and the *F1 score* which computes the harmonic mean of the latter two metrics. For completion, the *specificity* (i.e. the ratio of correctly predicted non-meteors) was also added, as in the future we plan to use the satellites in this set, for the context of SST. Within a completely balanced data set, the *recall* = *specificity*.

In view of the final results, the *Random Forest* model performed slightly better than the rest, obtaining the highest accuracy (98.2 per cent), followed closely by other ensemble methods (i.e. *Gradient Boost* and *Stacking Estimators*), which obtained a high score of 98 per cent for binary classification. Part of the poor classification can be explained by the imbalance between the classes, which leads the models to regard the meteor class as less representative. This can be observed in the confusion matrix (Fig. 11), as the real meteor predictions score is slightly lower (in general), compared to the non-meteor (TN) classification. Few data augmentation techniques were considered to balance the classes, however, due to the severe distortion of the all-sky lens, these would modify the detection physics. Thus, we decided to use only the unaltered data of the events.

The performance metrics of each model are displayed in Fig. 12. Among these, the most relevant one for meteor classification is the *recall*, as we want to keep as many real meteors as possible. Additionally, the meteor prediction branching (leakage) was computed (also using 10 randomized test results). This is defined by $B_m = FP/TP$ (the smaller the better). A leakage score of 0.10 (i.e. the one obtained with *Neural Net*), means that from all the meteors returned as True by the prediction, a proportion of 10 per cent will be non-meteors (FP). This is expected, due to the large difference between the number of true and false events. When we consider the *Recall* alone, we find that the *Ridge* model performs the best (Fig. 12), resulting in only a 3.3 per cent of true meteors being falsely discarded. However, it's accuracy score is among the lowest of the models (78 per cent), and this ensures a large number of FP slipping into the downstream trajectory computations, making the *Recall* alone not a useful metric. When both *Recall* and the leakage are taken into account, the *Neural Net* performs the best considering our requirement that a model should return a high real meteor rate (*Recall* = 95.8 per cent), with low number of FP (leakage = 10 per cent). When the multistation spatiotemporal coincidence threshold was applied before the ML step, a large improvement in both leakage (i.e. from 10 per cent $\rightarrow$ 0.2 per cent) and *Recall* (i.e. from 95.8 per cent $\rightarrow$ 99.92 per cent) scores was observed. This is comparable with the results of (Gural 2019), as the author also obtained a low leakage (0.4 per cent), however on a much larger data set. The spatiotemporal filter limits true detection only to those which were seen on at least two close stations (less than 600 km apart), and on a time gap of maximum 3 s between detections, timed by a Network Time Protocol. This coincidence improvement can be applied to events which were recorded on two stations or more, which in the case of MOROI, this happens on 20 per cent of the detections. Nonetheless, these are the events for which the trajectory and the orbit can be computed. When an altitude threshold is applied for the trajectories (i.e. starting height of $<$150 km, and end height of $>$20 km), the leakage drops to 0.0 per cent.

During the supervised validation, multiple events displayed normal meteor behaviour in the detection frames, despite the poor quality of the initial centroid estimates coming from FREETURE. Random loading of CPU cores during background tasks executions would cause a frame to be dropped, and some meteors have lost $>$60 per cent of the frames. Unusual data also appeared due to heavy extinction in the meteor line of sight (i.e. due to trees or spots on the camera dome), resulting in either incomplete trajectory or with large *rms* on initial positions; better quality could be achieved if the camera dome is clean (i.e. no dust, snow, ice, dew or scratches), or when there aren't any interruptions in detection sequence due to cloud coverage. Some of this behaviour is similar to a fragmenting meteor, however, in most of these subpar scenarios, the event displays behaviour which could not be found in fragmenting meteors either due to the regularity of interruptions, or the large difference between meteor positions (in the case of a localized dome absorption). These events can be filtered from the real meteors by the user, before conducting a future study for single-station photometry.

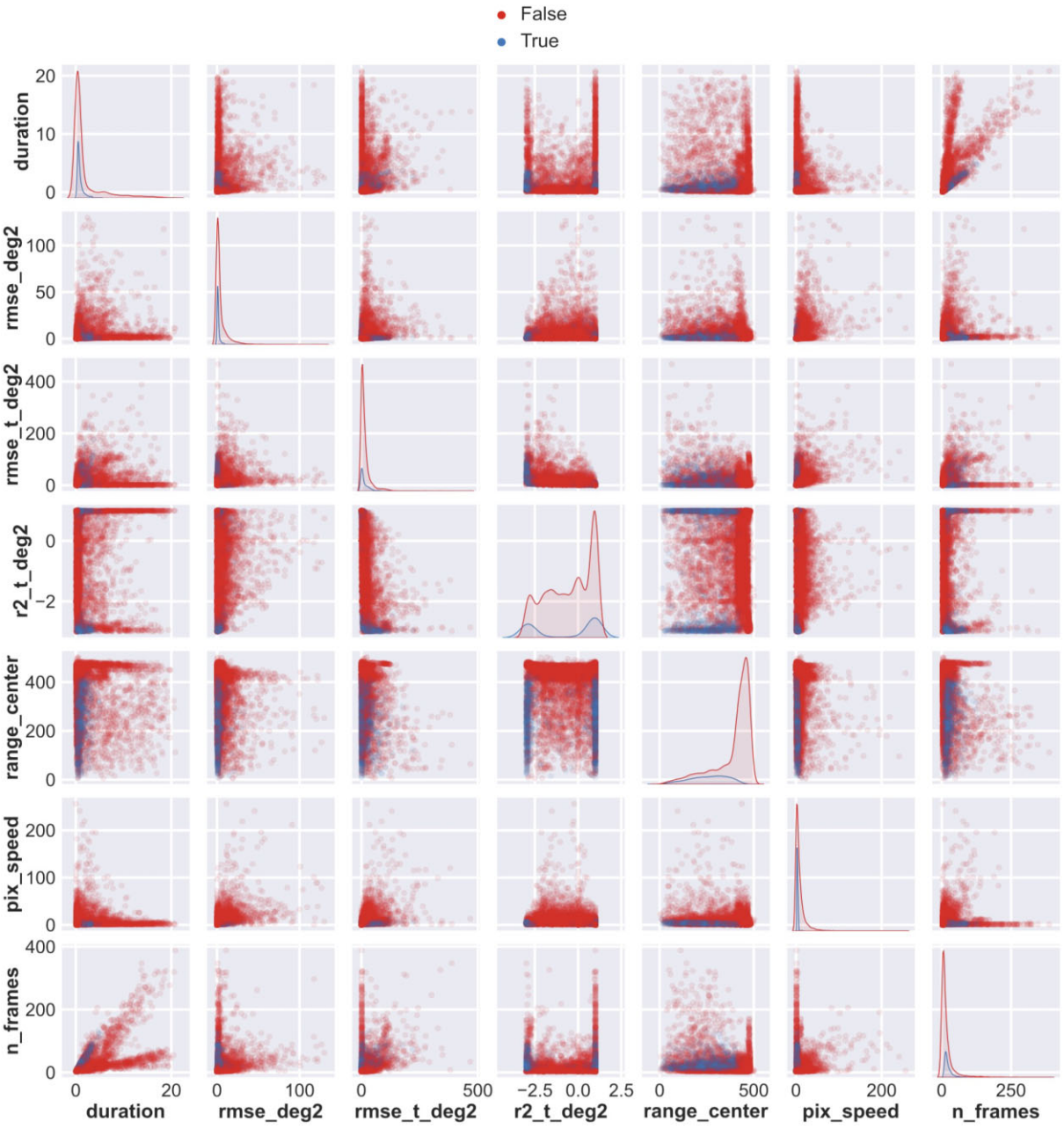

Figure 10. A pair plot mosaic of the 7 features used for the final score, to illustrate their dependence. The name *in-extenso* for the list of features can be found in Section 2.2. Each plot contains 6922 false (red) and 1164 true (blue) events. On the diagonal, the feature counts are displayed in a histogram-like function.

4.2 User validation bias

During data labelling, from time to time it proved quite challenging to distinguish between meteors and non-meteors. Thus additional data had to be inspected, before assigning the proper label to an event. However on limit examples, we estimate the labelling error of roughly 0.1 per cent, which would explain some of the uncertainty of the classifiers. Moreover, during the classification process, if there is only one station which observed the event, it will introduce a bias towards validating mostly events which show a somewhat typical meteor behaviour (i.e. light curve shape, duration, angular velocity), which may vary based on the user's experience in meteor observing. While meteors exhibit various ablation characteristics and morphologies, as discussed by e.g. Subasinghe & Campbell-Brown (2018), a better approach implies to keep as a valid ground truth training set only meteors which have been already validated by spatiotemporal observation. This method can be approached when the routines are applied to a larger data base (as in the case of FRIPON consortium), thus keeping the number of valid examples relevant for robust model training.

4.3 Score improvement ideas

For this study we aimed to train the models with only the features computed from the events positions found in `positions.txt` file, generated by FreeTure. Further improvements to the score can be attained by:

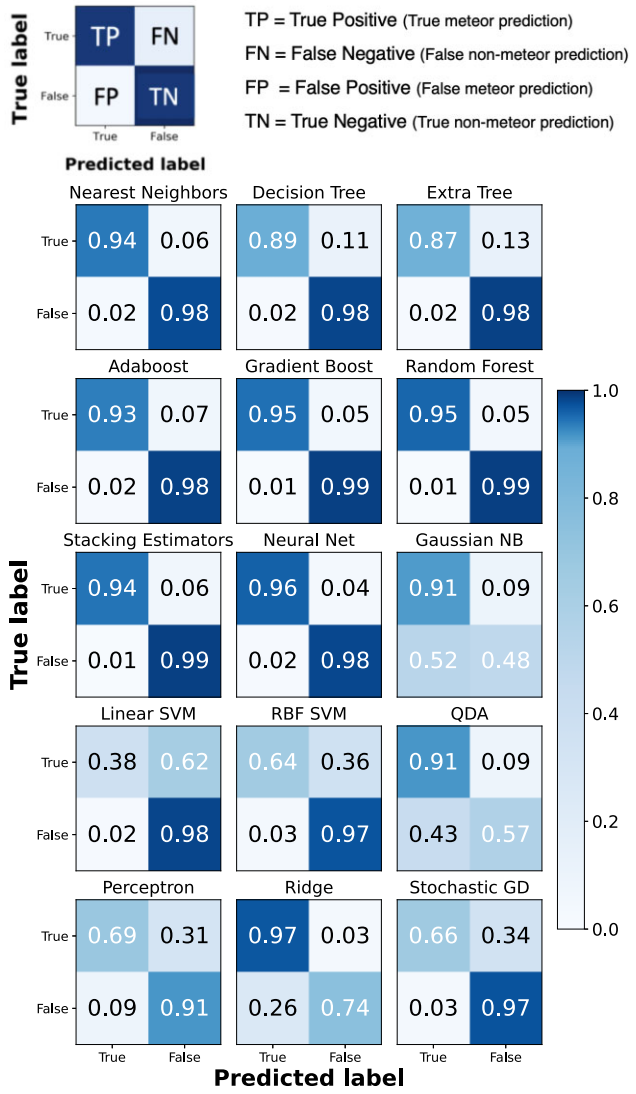

Figure 11. Top: The legend displaying the matrix grid labelling. Bottom: The confusion matrices each model trained using the top score features. The classification results are averaged over 10 runs, normalized and displayed as a heat map. Each box represents the classification ratio of the test data.

(i) setting a horizon mask designed for each station. This would limit the input of false detections. However, this would also remove true meteors close to the horizon. When this mask is avoided, the detection procedure generates an enhanced *working set,* which increases the number of examples used for training. Since our number of examples was not large, the mask was not used.

(ii) extracting the light sources from each frame, and use the ROI kernel attributes, as features for model training.

(iii) coupling the position of the centroid given by the FREETURE software, with the associated luminosity of the centroid, extracted from frames. The features presented in Section 2.2 do not include a luminosity aspect of the meteor. Such an extra feature could increase the score on meteor detection, since the non-meteors display a quite varied luminosity profile, ranging from a regular alternative luminosity, to a straight uniform line (in the case of satellites). This method also has applications in satellite detection from telescope images (Piso et al. 2021).

5 SUMMARY AND CONCLUSIONS

A supervised validation was applied to the MOROI event data base (N = 9858) recorded between 2017–2020. Among these:

(i) 1450 events were discarded due to non-meteor behaviour (e.g. cosmic rays, hot pixels), having one or two frames.

(ii) 8086 events (the *working set*), representing 1164 True meteors, and 6922 non-meteors (e.g. aeroplanes, satellites, city lights).

From a set of 24 features computed for each detection, 7 were found relevant for an increase in classification accuracy. These include the duration of the detection, the frame number, the speed of the detection in (pixels/frame), the range from the image centre, the RMSE of the polynomial fit, and both the RMSE and r^2 of the transposed polynomial fit. This list was obtained via implementing a *keep-best* feature cycling algorithm. For the model implementation, 15 ML classifiers were selected and tested for their ability to correctly attribute detections into either a meteor or a non-meteor.

The results show an average classification accuracy of 98.1 per cent (±0.4 per cent), obtained by Random Forest, Gradient Boost, Stacking Estimators algorithms. This is slightly superior than the accuracy obtained by an experienced human validator.

A more important metric is the *Recall* (i.e. the ratio of true meteors classified as true), for which the multilayer perceptron model (having 16 layers of 64 neurons each) obtained a score of 95.8 per cent. When a spatiotemporal coincidence filter was applied before the ML step (i.e. applicable on double- to multistation events), the *Recall* raised to 99.92 per cent, with a final leakage of 0.2 per cent.

We anticipate to further improve the model performance with complementary classification criteria, such as the meteor lightcurve, local sky background. Additionally, by balancing and increasing the training set (i.e. using the FRIPON data base) would only increase the model performance by allowing the classifiers to generalize using more examples.

This paper introduced a set of ML-based tools for recovering a subset of scientifically exploitable data otherwise lost during operation of wide-area, meteor detection networks. Single-station detections, usually discarded from further analysis, still contains valuable photometric and astrometric data that can constrain the nature of sporadic meteors.

In the future, we will train the models on the validated multistation events in the FRIPON data base, which will significantly increase the number statistics.

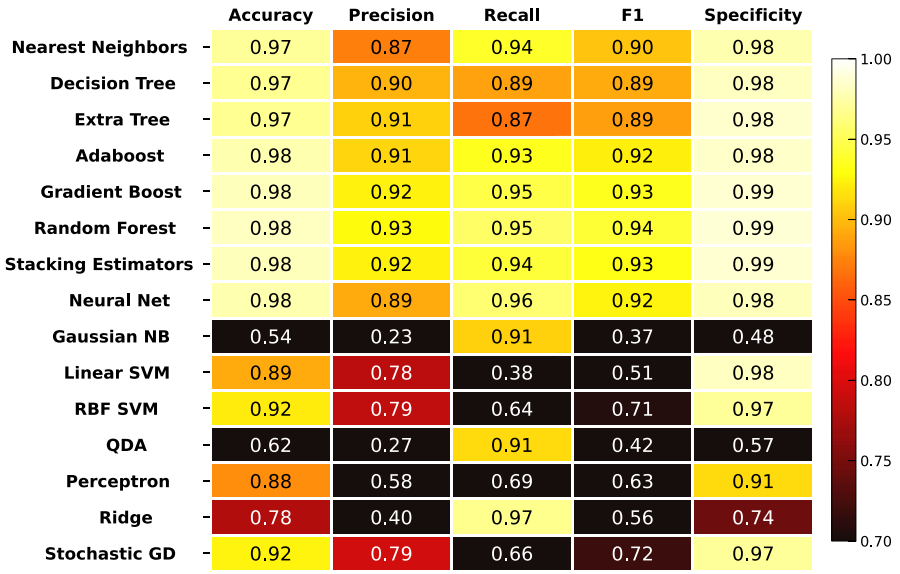

Figure 12. The classification results computed for all models. The scores are averaged over a total of 10 runs and displayed as a heat map. A higher precision of the full set of scores is displayed in Table B2.

ACKNOWLEDGEMENTS

SA and DAN were supported by a grant of the Romanian Ministery of Education and Research, CCDI - UEFISCDI, project number PN-III-P2-2.1-PTE-2019-0554, within PNCDI III. The work of IB and MB was supported by a grant of the Romanian Ministry of Research, Innovation and Digitalization, CNCS-UEFISCDI, CNCS-UEFISCDI, project number PN-III-P1-1.1-PD-2019-0784, within PNCDI III.

DATA AVAILABILITY

The data underlying this article are available in the article and in its online supplementary material, under the name : `events.txt`.

REFERENCES

Altman N. S., 1992, Am. Stat., 46, 175
Anghel S., Birlan M., Nedelcu D. A., Boaca I., 2019, Rom. Astron. J., 29, 189
Anghel S., Szücs-Csillik I., Nedelcu D. A., Boaca I., Birlan M., 2021a, Rom. Astron. J., 31, 153
Anghel S., Nedelcu D. A., Birlan M., Boaca I., Colas F., Malgoyre A., Moroi Fripon Teams, 2021b, in 84th Annual Meeting of the Meteoritical Society, Abs. 6027
Anghel S. et al., 2021c, MNRAS, 508, 5716
Audureau Y. et al., 2014, in Rault J. L., Roggemans P., eds, Proceedings of the International Meteor Conference, The International Meteor Organization, Giron, France, 18-21 September 2014. p. 39
Bannister S. M., Boucheron L. E., Voelz D. G., 2013, PASP, 125, 1108
Birlan M. et al., 2021, Rom. Astron. J., 31, 41
Breiman L., Friedman J., Stone C. J., Olshen R. A., 1984, Classification and Regression Trees. Chapman & Hall/CRC Press, Boca Raton, FL
Carass A. et al., 2020, Sci. Rep., 10, 8242
Chang C.-C., Lin C.-J., 2011, ACM Trans. Intell. Syst. Technol. (TIST), 2, 1
Colas F. et al., 2020, A&A, 644, A53
Cooke W. J., Moser D. E., 2012, in Proceedings of the International Meteor Conference, 30th IMC, Sibiu, Romania, 2011. p. 9
De Cicco M. et al., 2018, in Gyssens M., Rault J.-L., eds, Proceedings of the International Meteor Conference, Petnica, Serbia, 2017. p. 65
De Bièvre P., 2012, Chemistry International–Newsmagazine for IUPAC, 34, 26
Emran S. M., Ye N., 2002, Qual. Reliab. Eng. Int., 18, 19
Freund Y., Schapire R. E., 1997, J. Comput. Syst. Sci., 55, 119
Freund Y., Schapire R. E., 1999, Mach. Learn., 37, 277
Friedman J. H., 2001, Ann. Stat., 29, 1189
Galindo Y., Lorena A. C., 2018, in Anais do XV Encontro Nacional de Inteligência Artificial e Computacional. SBC, KS 66202, p. 528
Gardiol D. et al., 2021, MNRAS, 501, 1215
Geurts P., Ernst D., Wehenkel L., 2006, Mach. Learn., 63, 3
Gural P. S., 2019, MNRAS, 489, 5109
Halliday I., Griffin A. A., Blackwell A. T., 1996, Meteorit. Planet. Sci., 31, 185
Hastie T., Tibshirani R., Friedman J., 2008, The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, New York
Hinton G. E., 1990, in Machine Learning. Elsevier, Amsterdam, Netherlands, p. 555
Hinton G. E., Salakhutdinov R. R., 2006, Science, 313, 504
Howie R. M., Paxman J., Bland P. A., Towner M. C., Cupak M., Sansom E. K., Devillepoix H. A., 2017, Exp. Astron., 43, 237
Jenniskens P., Gural P. S., Dynneson L., Grigsby B. J., Newman K. E., Borden M., Koop M., Holman D., 2011, Icarus, 216, 40
Krizhevsky A., Sutskever I., Hinton G. E., 2012, Adv. Neural Inf. Process. Syst., 25, 1097
LeCun Y., Bottou L., Bengio Y., Haffner P., 1998, Proc. IEEE, 86, 2278
Nedelcu D. A. et al., 2018, Rom. Astron. J., 28, 57
Pedregosa F. et al., 2012, preprint (arXiv:1201.0490)
Peña-Asensio E. et al., 2021, European Planetary Science Congress. p. EPSC2021–738
Piso A.-M. A., Voicu O.-A., Sprimont P.-G., Bija B., Lasheras O. A., 2021, in Flohrer T., Lemmens S.,Schmitz F., eds, 8th European Conference on Space Debris, vol. 8. European Space Agency
Rennie J. D., Shih L., Teevan J., Karger D. R., 2003, in Fawcett T., Mishra N., eds, Proceedings of the 20th International Conference on Machine Learning (ICML-2003), Washington DC, 2003. The AAAI Press, California, p. 616
Rifkin R. M., Lippert R. A., 2007, Computer Science and Artificial Intelligence Laboratory Technical Report, MIT, MA 02139
Rumelhart D. E., Hinton G. E., Williams R. J., 1986, Nature, 323, 533
Silađi E., Vida D., Nyarko E. K., 2015, in Proceedings of the International Meteor Conference. Mistelbach, Austria, p. 27
Stehman S. V., 1997, Remote Sens. Environ., 62, 77
Subasinghe D., Campbell-Brown M., 2018, AJ, 155, 88
Towner M. C. et al., 2020, PASA, 37, e008

Vida D., Zubović D., Šegon D., Gural P., Cupec R., 2016, in Roggemans A., Roggemans P., eds, International Meteor Conference Egmond, the Netherlands, 2-5 June 2016. International Meteor Organization, p. 307
Vida D., Blaauw Erskine R. C., Brown P. G., Kambulow J., Campbell-Brown M., Mazur M. J., 2022, MNRAS, 515, 2322
Weisstein E. W., 2022, Statistical Correlation, https://mathworld.wolfram.com/StatisticalCorrelation.html, (last accessed 2022.11.22)
Weryk R. J., Brown P. G., Domokos A., Edwards W. N., Krzeminski Z., Nudds S. H., Welch D. L., 2008, Earth Moon Planets, 102, 241
Wolpert D. H., 1992, Neural Netw., 5, 241
Yerushalmy J., 1947, Public Health Reports (1896-1970), Sage Publications, Inc., p. 1432

SUPPORTING INFORMATION

Supplementary data are available at *MNRAS* online.

events.txt

Please note: Oxford University Press is not responsible for the content or functionality of any supporting materials supplied by the authors. Any queries (other than missing material) should be directed to the corresponding author for the article.

APPENDIX A: MODEL PERFORMANCE METRICS

In this section, several performance metrics are presented, along with an application example using the data points kept for testing (N = 2426). The equation list includes:

A1 Accuracy or classification accuracy

Used in classification problems, the *accuracy* (De Bièvre 2012) of a predictive model refers to the ratio of correct predictions to the rest of the predictions. The expression can be written as:

$$\begin{aligned} Accuracy &= \frac{\#\,correctly\ predicted\ data}{\#\,total\ testing\ data} \\ &= \frac{TP + TN}{TP + TN + FP + FN} \\ &= \frac{201 + 1956}{201 + 1956 + 131 + 138} \times 100\ \text{per cent} \\ &= 88.9\ \text{per cent}, \end{aligned} \tag{A1}$$

where 88.9 per cent is the accuracy (i.e. how close is the prediction from the ground truth), obtained with the meteor test set, in one of the computation runs. This approach is relevant when the two variable classes in the data are closely balanced. In our case, the non-meteor to meteor ratio is 5:1, thus, the accuracy diminishes a significant proportion of *type 1 errors* and is not a proper evaluation metric. Therefore, we are computing a set of additional metrics to evaluate the models.

A2 Precision

Precision is a common method to measure the closeness of agreement between results (De Bièvre 2012). In this context, the *precision* measures the ratio of the correctly predicted meteors, to the the total number of meter predictions. The relation goes as:

$$\begin{aligned} Precision &= \frac{\#\,correctly\ predicted\ true\ meteors}{\#\,total\ meteor\ predictions} \\ &= \frac{TP}{TP + FP} \\ &= \frac{201}{201 + 131} = 0.61. \end{aligned} \tag{A2}$$

Here, 0.61 translates, when the model predicts that an event has meteor behaviour, it is correct in 61 per cent of the time. Thus, it is also known as the Positive Predictive Value.

A3 Recall or sensitivity

Recall or hit rate (Yerushalmy 1947) measures what proportion of events that are actual meteors, were also predicted to be meteors. It is also called *sensitivity* as it answers the following question: 'How sensitive is the model in detecting actual meteor events?'

$$\begin{aligned} Recall &= \frac{\#\,correctly\ predicted\ true\ meteors}{\#\,total\ true\ meteors} \\ &= \frac{TP}{TP + FN} \\ &= \frac{201}{201 + 138} = 0.59, \end{aligned} \tag{A3}$$

where 0.59 means that of the actual true meteor events, 59 per cent are correctly predicted by the classifier to be true. In statistical classification, it is also known as True Positive Rate.

A4 F1 score

Also known as the Sørensen–Dice coefficient, the *F1 score* is the harmonic mean of the precision and recall. For more information, the reader is referred to Carass et al. (2020) and the references therein. As well as for the other metrics used in this study, the score range between 0 and 1.0. *F1 score* can start from 0 (if either *precision* or *recall* is 0) and reach a maximum possible score of 1, which indicates a perfect match on both precision and recall. The relation can be written as:

$$\begin{aligned} F1\ Score &= \frac{2 \times Precision \times Recall}{Precision + Recall} \\ &= \frac{2 \times 0.61 \times 0.59}{0.61 + 0.59} = 0.6. \end{aligned} \tag{A4}$$

A high *F1 score*, indicates good classification ratio for both precision and recall results.

A5 Specificity or selectivity

Also known as the True Negative Rate (Yerushalmy 1947), it measures the ratio of non-meteors that were correctly predicted.

$$\begin{aligned} Specificity &= \frac{\#\,correctly\ predicted\ non-meteors}{\#\,total\ non-meteors} \\ &= \frac{TN}{TN + FP} = \frac{1956}{1956 + 131} = 0.94. \end{aligned} \tag{A5}$$

A specificity or selectivity of 0.94 represents 94 per cent of false events (non-meteors) were predicted as non-meteors.

For the final results, we employ the metrics presented here, to compare all the models used in this analysis.

APPENDIX B: MODEL SCORES

Table B1. Hyper-parameter tuning results.

Classifier name	Hyper-parameter	Range of tested values	Best value
Nearest Neighbors	n_neighbors	[1–60]	18
	algorithm	[‘auto’, ‘ball_tree’, ‘brute’]	‘auto’
	metric	[‘cityblock’, ‘euclidean’, ‘l2’, ‘l1’, ‘dice’, ‘sokalmichener’, ‘jaccard’, ‘kulsinski’, ‘matching’, ‘minkowski’, ‘hamming’, ‘rogerstanimoto’, ‘manhattan’, ‘canberra’, ‘russellrao’, ‘sokalsneath’]	‘canberra’
	weights	[‘uniform’, ‘distance’]	‘distance’
	leaf_size	1-60	27
Decision Tree	criterion	[‘gini’, ‘entropy’]	gini
	splitter	[‘best’, ‘random’]	best
	max_depth	[None, 10, 30, 50, 70, 90, 110]	30
	min_samples_split	[2, 3, 4]	2
	min_samples_leaf	[1, 2, 3]	1
	min_weight_fraction_leaf	[0.-0.5]	0
	max_features	[0.1, 0.5, 1, 2, ‘auto’, ‘sqrt’, ‘log2’]	1
	max_leaf_nodes	[2-10]	10
	ccp_alpha	[0-2]	0
Extra Tree	criterion	[‘gini’, ‘entropy’]	gini
	splitter	[‘best’, ‘random’]	best
	min_samples_split	[2-5]	3
	min_samples_leaf	[1 - 20]	12
	min_weight_fraction_leaf	[0 - 0.5]	0
	max_features	[0.1, 0.5, 1, 2, ‘auto’, ‘sqrt’, ‘log2’]	0.5
Adaboost	base_estimator	DecisionTreeClassifier	DecisionTreeClassifier
	learning_rate	[0.01-10]	0.22
	n_estimators	[1 - 100]	83
Gradient Boost	loss	[‘deviance’, ‘exponential’]	deviance
	learning_rate	[0.001-100]	0.1
	n_estimators	[1-250]	207
	subsample	[0.1-10]	0.1
	criterion	[‘friedman_mse’, ‘mse’]	friedman_mse
	min_samples_leaf	[1-30]	20
Random Forest	n_estimators	[10-80]	17
	max_features	[‘auto’, ‘sqrt’]	auto
	max_depth	[2, 4]	4
	min_samples_split	[2, 3, 4, 5]	2
	min_samples_leaf	[1, 2, 3, 4]	2
	bootstrap	[True, False]	False
Stacking Estimators	final_estimator	[None, LogisticRegression()]	None
	cv	[None, ‘StratifiedKFold’]	None
	stack_method	[‘auto’, ‘predict_proba’, ‘decision_function’, ‘predict’]	auto
	passthrough	[False, True]	False
Neural Net	hidden_layer_sizes	8-32 layers of 16-128 neurons	16 layers of 64 neurons
	activation	[‘relu’, ‘logistic’, ‘tanh’, ‘identity’]	relu
	solver	[‘adam’, ‘lbfgs’, ‘sgd’]	adam
	alpha	[0.00001 - 0.001]	0.00001
	batch_size	[‘auto’, 32, 64, 128, 256]	64
	learning_rate	[‘constant’, ‘invscaling’, ‘adaptive’]	constant
	learning_rate_init	[0.00001 - 0.01]	0.0001
	max_iter	[500, 1000, 1500]	500
	early_stopping	[False, True]	False
	epsilon	[1.e-10 - 1.e-06]	1.e-08
	warm_start	[False, True]	False
Gaussian NB	var_smoothing	[1.e-10 - 1.e + 10]	1.e-2
Linear SVM	loss	[‘squared_hinge’, ‘hinge’]	squared_hinge
	C	[1.e+02 - 1.e + 06]	1.e + 3
	multi_class	[‘ovr’, ‘crammer_singer’]	ovr
	fit_intercept	[True, False]	True
	intercept_scaling	[1.e+00 - 1.e + 04]	1.e + 1
	class_weight	[None, ‘balanced’]	balanced

Table B1 *– continued*

Classifier name	Hyper-parameter	Range of tested values	Best value
RBF SVM	C	[1.e-02 - 1.e + 05]	1.e + 04
	gamma	[1.e-06 - 1.e + 02, 'scale', 'auto']	1.e-05
	coef0	[0.1 - 100]	0.1
	shrinking	[True, False]	True
	cache_size	[100, 400]	100
	class_weight	[None, 'balanced']	None
	max_iter	[0 - -3]	-1
	break_ties	[False, True]	False
Perceptron	alpha	[1.e-10 - 1.e-04]	1.e-10
	l1_ratio	[0.01 - 0.2]	0.01
	fit_intercept	[True, False]	False
	max_iter	[200, 500, 1000]	200
	shuffle	[True, False]	False
	eta0	[0.01 - 2]	0.01
	validation_fraction	[0.01 - 0.9]	0.01
	n_iter_no_change	[5 - 25]	20
	warm_start	[False, True]	False
	early_stopping	[False, True]	False
Ridge	alpha	[1.e+00 - 1.e + 10]	1.e + 07
	fit_intercept	[True, False]	False
	normalize	[True, False]	True
	copy_X	[True, False]	False
	max_iter	[None, 'balanced']	balanced
	tol	[1.e-10 - 1.e + 05]	1e + 02
	solver	['auto', 'svd', 'cholesky']	auto
Stochastic GD	loss	['hinge', 'log', 'modified_huber', 'squared_hinge', 'perceptron', 'squared_loss', 'huber', 'epsilon_insensitive', 'squared_epsilon_insensitive']	epsilon_insensitive
	fit_intercept	[True, False]	False
	shuffle	[True, False]	False
	epsilon	[1.00000e-01 - 1.00000e + 04]	1. + 1
	eta0	[0 - 0.7]	0.25
	early_stopping	[False, True]	True

Table B2. The classification results computed for all models. All scores represent the average of 10 randomized sessions of training and testing on the working set.

Model Name	Accuracy	Precision	Recall	F1	Specificity
Nearest Neighbors	0.970	0.872	0.938	0.904	0.976
Decision Tree	0.968	0.895	0.887	0.891	0.982
Extra Tree	0.967	0.908	0.867	0.887	0.985
Adaboost	0.977	0.911	0.933	0.922	0.984
Gradient Boost	0.980	0.923	0.946	0.934	0.986
Random Forest	0.982	0.927	0.954	0.941	0.987
Stacking Estimators	0.980	0.922	0.942	0.932	0.986
Neural Net	0.977	0.891	0.958	0.924	0.980
Gaussian NB	0.544	0.233	0.908	0.371	0.480
Linear SVM	0.892	0.783	0.375	0.507	0.982
RBF SVM	0.921	0.786	0.642	0.706	0.970
QDA	0.621	0.270	0.912	0.416	0.570
Perceptron	0.879	0.579	0.688	0.629	0.913
Ridge	0.777	0.397	0.967	0.563	0.745
Stochastic GD	0.924	0.794	0.658	0.720	0.970

This paper has been typeset from a TEX/LATEX file prepared by the author.