Astrophys Space Sci (2018) 363:197
https://doi.org/10.1007/s10509-018-3418-7

ORIGINAL ARTICLE

Protostellar classification using supervised machine learning algorithms

O. Miettinen[1]

Received: 12 April 2018 / Accepted: 24 August 2018 / Published online: 29 August 2018

Abstract Classification of young stellar objects (YSOs) into different evolutionary stages helps us to understand the formation process of new stars and planetary systems. Such classification has traditionally been based on spectral energy distribution (SED) analysis. An alternative approach is provided by supervised machine learning algorithms, which can be trained to classify large samples of YSOs much faster than via SED analysis. We attempt to classify a sample of Orion YSOs (the parent sample size is 330) into different classes, where each source has already been classified using multiwavelength SED analysis. We used eight different learning algorithms to classify the target YSOs, namely a decision tree, random forest, gradient boosting machine (GBM), logistic regression, naïve Bayes classifier, *k*-nearest neighbour classifier, support vector machine, and neural network. The classifiers were trained and tested by using a 10-fold cross-validation procedure. As the learning features, we employed ten different continuum flux densities spanning from the near-infrared to submillimetre wavebands ($\lambda = 3.6$–870 μm). With a classification accuracy of 82% (with respect to the SED-based classes), a GBM algorithm was found to exhibit the best performance. The lowest accuracy of 47% was obtained with a naïve Bayes classifier. Our analysis suggests that the inclusion of the 3.6 μm and 24 μm flux densities is useful to maximise the YSO classification accuracy. Although machine learning has the potential to provide a rapid and fairly reliable way to classify YSOs, an SED analysis is still needed to derive the physical properties of the sources (e.g. dust temperature and mass), and to create the labelled training data. The machine learning classification accuracies can be improved with respect to the present results by using larger data sets, more detailed missing value imputation, and advanced ensemble methods (e.g. extreme gradient boosting). Overall, the application of machine learning is expected to be very useful in the era of big astronomical data, for example to quickly assemble interesting target source samples for follow-up studies.

Keywords Methods: data analysis · Stars: formation · Stars: protostars

✉ O. Miettinen
oskari.miettinen@digia.com

1 Digia Plc/Avarea Oy, Rautatieläisenkatu 6, 00520 Helsinki, Finland

1 Introduction

An essential part of the star formation studies is to try to classify the young stellar objects (YSOs) into different evolutionary stages, and construct a coherent YSO evolutionary sequence. Also, by determining the relative percentages of YSOs in different stages, the statistical time spent in each stage can be constrained, which in turn helps to quantify the overall timescale of the stellar birth process in different molecular cloud environments (e.g. Evans et al. 2009; Dunham et al. 2015).

Considering the formation of low-mass, solar-type stars, the YSOs have traditionally been classified into distinct stages on the basis of their infrared (IR) spectral slopes (e.g. Lada and Wilking 1984) or bolometric temperatures (Myers and Ladd 1993). In particular, the spectral energy distribution (SED) of a YSO, which is characterised by the bolometric temperature and luminosity, is commonly used to determine the evolutionary stage of the source, that is whether it is a so-called Class 0 or I protostar, or Class II or III pre-main sequence (PMS) star (e.g. Lada 1987; Adams et al. 1987; André et al. 1993; see also André et al. 2000 for a review). Indeed, an SED analysis is very useful, not just for the purpose of source classification, but to derive some of the key physical properties of the source, such as the dust temperature and dust mass. However, modelling the source SEDs can be fairly time consuming, and hence, to quickly determine the evolutionary classes for a large sample of YSOs, an automated procedure that employs the observed source properties (i.e. the flux densities) would be very useful. In this regard, machine learning has the potential to yield a fast way to classify sources (as compared to an SED analysis) as long as the algorithm(s) in question can be trained with data sets composed of relevant flux densities and corresponding evolutionary classes of the target YSOs.

So far, machine learning based classification of astrophysical objects has mostly been applied in extragalactic research (e.g. Krakowski et al. 2016; Aniyan and Thorat 2017; Sreejith et al. 2018; Beck et al. 2018; Pashchenko et al. 2018; Hui et al. 2018; Lukic et al. 2018; An et al. 2018; see also Lochner et al. 2016), while Galactic machine learning studies have been relatively few in number (e.g. Marton et al. 2016; Yan et al. 2018). Hence, pilot studies about using machine learning in YSO classification, which the present work represents, are warranted.

In this paper, we report the results of our protostellar classification test using several different supervised machine learning algorithms. The data set used in this study is described in Sect. 2, while the data analysis is presented in Sect. 3. The results are presented and discussed in Sect. 4, and in Sect. 5 we summarise the key results and conclusions of this work.

2 Data

The data analysed in this paper were taken from Furlan et al. (2016, hereafter FFA16). As part of the *Herschel*[1] Orion Protostar Survey (HOPS; e.g. Stutz et al. 2013), FFA16 studied and modelled the SEDs of a large, homogeneous sample of 330 YSOs in the Orion molecular cloud complex (the authors assumed a uniform distance of 420 pc to the cloud complex). At the time of writing, this is the largest available YSO sample investigated in a single star-forming cloud complex. The photometric data employed by the authors included the J, H, and K_S near-IR data from the Two Micron All Sky Survey (2MASS; Skrutskie et al. 2006), *Spitzer* IR data obtained with the Infrared Array Camera (IRAC; 3.6–8.0 µm; Fazio et al. 2004), the Multiband Imaging Photometer for *Spitzer* (MIPS; 24 µm; Rieke et al. 2004), and the Infrared Spectrograph (IRS; 5.4–35 µm; Houck et al. 2004). The *Herschel* satellite (Pilbratt et al. 2010) was used to observe the 70, 100, and 160 µm far-IR bands with the Photodetector Array Camera and Spectrometer (PACS; Poglitsch et al. 2010), while the 350 and 870 µm submillimetre data were obtained with the Atacama Pathfinder EXperiment (APEX; Güsten et al. 2006) using its Submillimetre APEX BOlometer CAmera (SABOCA; Siringo et al. 2010) and Large APEX BOlometer CAmera (LABOCA; Siringo et al. 2009). The aforementioned bands cover the typical protostellar SED peak emission at ~ 100 µm and its surrounding wavelengths, which is essential to determine reliable physical properties and evolutionary stage of the source (see below).

On the basis of their panchromatic SED analysis, FFA16 classified their YSO sample into 92 Class 0 protostars, 125 Class I protostars, 102 flat-spectrum sources (expected to be objects in transition between the Class I and II phases), and 11 Class II PMS stars. The corresponding relative percentages are $27.9\% \pm 2.9\%$, $37.9\% \pm 3.4\%$, $30.9\% \pm 3.1\%$, and $3.3\% \pm 1.0\%$, respectively, where the quoted uncertainties represent the Poisson counting errors. We note that one of the FFA16 sources, namely the Class 0 source HOPS 400, was first uncovered by Miettinen et al. (2009; their source SMM 3) in their LABOCA imaging of the Orion B9 star-forming region (see also Miettinen 2016, and references therein).

In brief, the physical explanation of why the SED analysis presented by FFA16 can be used to classify YSOs into different evolutionary stages is as follows (see e.g. White et al. 2007; Dunham et al. 2014 for reviews). The youngest protostellar objects, or Class 0 objects, are characterised by a central protostar deeply embedded in its cold, dusty envelope (André et al. 1993; André and Montmerle 1994). Hence, the source is extremely faint in the optical ($\lambda \sim$ 0.4–0.7 µm) and near-IR ($\lambda \gtrsim 0.7$–5 µm; traced by 2MASS and *Spitzer*/IRAC observations), but bright in the far-IR ($\lambda \sim 25$–350 µm; traced by *Spitzer*/MIPS and *Herschel* observations) and (sub-)millimetre ($\lambda \gtrsim 350$–1 000 µm; traced by APEX bolometer observations) dust emission. The central protostar increases its mass by accreting gas from the surrounding envelope via a circumstellar disk. When the envelope mass has dropped to that of the growing central protostar, the system is believed to transition from the Class 0 to the Class I stage. Class I protostars are still surrounded by an accretion disk and a circumstellar envelope of gas and dust, and hence their SEDs peak in the far-IR. However, another prominent SED bump can be seen in the mid-IR ($\lambda \sim 5$–25 µm; traced by *Spitzer*/IRAC and MIPS observations), which is an indication of hotter dust than in the previous Class 0 stage. If the central protostar can be seen along the long axis of the protostellar outflow, the Class I object can be optically visible. The intermediate stage between the Class I and Class II stages is characterised by the disappearing dust excess emission in the mid-IR, and hence the sources in this transition stage are known as flat-spectrum (SED) sources (Greene et al. 1994). In the Class II stage, the envelope has dissipated, and an optically visible PMS star is

[1] *Herschel* is an ESA space observatory with science instruments provided by European-led Principal Investigator consortia and with important participation from NASA.

surrounded by a tenuous disk. The SEDs of Class II objects peak at visible or near-IR wavelengths, and the disk adds an IR excess to the SED.

3 Data analysis

3.1 Source selection

One of the four YSO classes from FFA16, namely the Class II phase, is highly unbalanced with respect to the other three classes (Class 0, Class I, and flat sources). Indeed, only 3.3% of the FFA16 sources were classified as Class II sources, which is about ten times less than the occurrence of other types of sources. Although techniques to deal with imbalanced data sets exist (e.g. Kotsiantis et al. 2006a for a review; He and Ma 2013), we do not consider the FFA16 Class II sources in the subsequent analysis because their relative rarity can lead to problems in the training of the supervised classification algorithms, and in the evaluation of the classifiers' performance.

After discarding the Class II sources, we are left with 319 sources, out of which 28.8% ± 3.0% are Class 0 sources, 39.1% ± 3.5% are Class I sources, and 32.0% ± 3.3% are flat-spectrum sources. Hence, all these three classes are in good relative number balance with respect to each other.

3.2 Missing value treatment

When developing machine learning models, it is important to handle the missing values in the analysed data set (e.g. Witten and Frank 2005; Saar-Tsechansky and Provost 2007). A common approach is to replace the missing values by the mean of the non-missing values for the variable or feature in question (e.g. Alpaydin 2010; Lantz 2015). This imputation method does not change the sample mean of the variable.

We found that 69.3%, 54.9%, and 40.1% of the selected FFA16 sources missed the 2MASS J, H, and K_S near-IR data. Owing to these large percentages, we discarded the 2MASS data in our subsequent analysis. Also, 31% of the selected FFA16 YSOs lacked the SABOCA 350 µm data, but we included this waveband in the analysis because it provides a useful data point in the Rayleigh-Jeans part of the source SED, and the band has also been covered by observations with other instruments, such as by *Herschel* submm surveys (e.g. André et al. 2010). All the remaining bands except the *Herschel* 70 µm band were also found to contain missing values, but in those cases only 0.3% to 16.6% (11% on average) of the sources had missing data.

To impute the missing values, we used the `R` package `MICE` (Multivariate Imputation via Chained Equations; van Buuren and Groothuis-Oudshoorn 2011). The usage of `MICE` is based on the assumption that the missing data are Missing at Random (MAR), and it imputes data on a variable by variable basis (i.e. flux density by flux density basis in our case) by specifying an imputation model per variable. To calculate the imputations, we used the predictive mean matching (PMM) method (Little 1988). If $S_{\rm miss}$ is the variable that contains missing data, and $S_{\rm i}$ are the variables that do not suffer from missing data, the PMM algorithm works as follows: i) for cases with no missing data, a linear regression model of $S_{\rm miss}$ is estimated on $S_{\rm i}$, which yields a set of coefficients b; ii) a random draw is taken from the distribution of b values, which yields a new sample of coefficients b^*; this step is needed to generate some random variability in the imputed values; iii) the b values are used to calculate the predicted values for the observed $S_{\rm miss}$ values, while the b^* values are used to calculate the predicted values for the missing $S_{\rm miss}$ values; iv) for each case with a missing $S_{\rm miss}$ value, a set of cases with $S_{\rm miss}$ present is identified where the predicted values are closest to the predicted value for the case with a missing $S_{\rm miss}$ value; v) from the latter cases, a random value is chosen, and it is then used to impute the missing value. The PMM method is expected to lead to more reasonable estimates of the missing flux densities than simply using the sample averages, while still being a very fast imputation method.

Regarding the traditional mean or median imputations, one might think that replacing the missing flux density values by the mean or median values for each protostellar class separately would be a better approach than using the full sample mean or median. Indeed, this would be a more physical approach than using the full sample mean or median imputation because the flux density at a given wavelength can evolve as the source evolves (e.g. the amount of dust in the protostellar envelope decreases as the source matures, which affects the far-IR and submm emission considered in the present study). However, to do this, one would have to use information from the test set's labels or classes, and one should not leak such information (which is correlated with the labels) to the training procedure of a classifier. More importantly, when analysing new, previously unseen data, the protostellar or YSO classes are not even known, but they are what one wants to determine or predict.

On the other hand, it is known that for example the far-IR and submm flux densities considered in the present work depend on each other via the frequency dependent dust opacity, $\kappa_\nu \propto \nu^\beta$, where β is the dust emissivity index (e.g. Shetty et al. 2009). Hence, the missing submm flux densities could also be estimated in a source-by-source fashion from the existing ones by assuming a value for β. However, estimating the missing flux density values this way requires more feature value engineering, and hence is not as fast as for example the PMM. Indeed, obtaining fast classifications (as compared to an SED analysis) is the key of applying machine learning in the first place.

3.3 Training and test data sets

A supervised machine learning algorithm requires a so-called training set through which the algorithm tries to learn how the input values, or features (flux densities in our case), map to the response values, or labels (protostellar classes in the present work). Another data set, the so-called test set, is then used to test the performance of the trained model by showing it only the input values, and see which corresponding classes the model predicts for the data it did not see during training.

A traditional way to create the training and test data sets is to fragment the original data into two parts. For example, one common way is to use 80% of the data for training the algorithm, and the remaining 20% for testing it (the so-called 80/20 rule or the Pareto principle; e.g. Box and Meyer 1986). However, the present data set is fairly small for a machine learning experiment (319 sources in total), and hence the aforementioned data splitting would not yield good training and test capabilities. Instead, we used the technique of k-fold cross-validation (CV), where the data set is randomly divided into k roughly equal sized subsamples, or folds (Mosteller and Turkey 1968; see e.g. James et al. 2017, Sect. 5.1.3 therein). The algorithm is trained using $k-1$ of the folds, and the resulting model is tested on the remaining part of the data. This procedure is then repeated k times. We did the sampling with replacement, which means that the same source could be sampled more than once. The final prediction performance was taken as the mean of the k results. A value of $k=10$ was used in the present work.

Another caveat of dealing with small data sets in machine learning is the problem of overfitting, which means that an algorithm starts to learn the details of the training data set, including the random noise features, such as outliers instead of (or besides) the underlying general rules, patterns, or relationships (e.g. Hawkins 2004). This weakens the algorithm's ability to generalise well to previously unseen cases, because the aforementioned noise features are unlikely to occur in the test and new data sets. However, the usage of k-fold CV can reduce the degree of overfitting (but not fully prevent it) owing to the split of the data into multiple training and test sets (e.g. Cawley and Talbot 2010).

3.4 Principal component analysis

Another data preprocessing step we did was principal component analysis (PCA; Pearson 1901; Hotelling 1933; Jolliffe 2002; Abdi and Williams 2010), which is a common technique in machine learning for extracting the most important variables among a large number of variables in a data set, and hence to overcome feature redundancy and reduce the dimensionality of the problem. We note that if the analysed data set is split into separate labelled training set and test set, then the PCs need to be calculated on the training data set. However, because in the present study we employed the method of k-fold CV for training and testing the classifiers, we carried out the PCA using the full data set (i.e. 319 sources). Because in the PCA the original data are projected onto directions that maximise the variance, the variables were scaled to have a variance equal to $\sigma_{\rm SD}^2=1$, where $\sigma_{\rm SD}$ is the standard deviation, before the PCs were calculated.

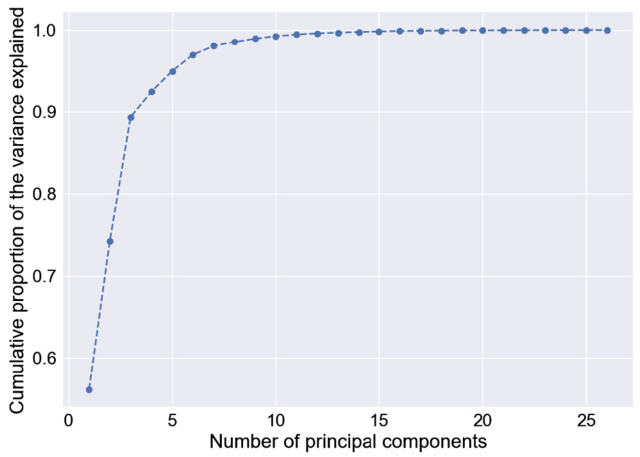


Fig. 1 Cumulative proportion of the variance in the data explained by each PC. The first ten PCs explain about 99.2% of the variance

In Fig. 1, we plot the cumulative proportion of the variance in the data set explained by each PC. In this analysis, all the other wavebands except those of 2MASS were taken into account (i.e. four *Spitzer*/IRAC bands, one *Spitzer*/MIPS band, three *Herschel*/PACS bands, two APEX bands, and 16 *Spitzer*/IRS bands). We found that the first ten PCs explain about 99.2% of the variance in the data, and hence we used only the first ten features in the subsequent analysis. These ten features correspond to the continuum flux densities in the FFA16 photometric data, that is from the *Spitzer*/IRAC 3.6 µm to LABOCA 870 µm data, while the *Spitzer*/IRS data were discarded. The distributions of the considered flux densities are presented in Fig. 2, which shows a draftsman's plot, or a scatter matrix plot (see e.g. NIST/SEMATECH e-Handbook of Statistical Methods[2]), and which enables to see the interrelations between variables in multivariate data (ten dimensional in our case). The plot consists of an array of two-variable scatter diagrams. For example, the plot demonstrates the strong correlation between the *Herschel* 100 µm and 160 µm flux densities.

Besides reducing the dimensionality of the problem, the feature selection enabled by PCA also helps to relieve the problem of overfitting discussed in Sect. 3.3. The reason for this is that the trained model will be less complex the smaller the number of features is relative to the number of

[2]https://www.itl.nist.gov/div898/handbook/eda/section3/scatterb.htm.

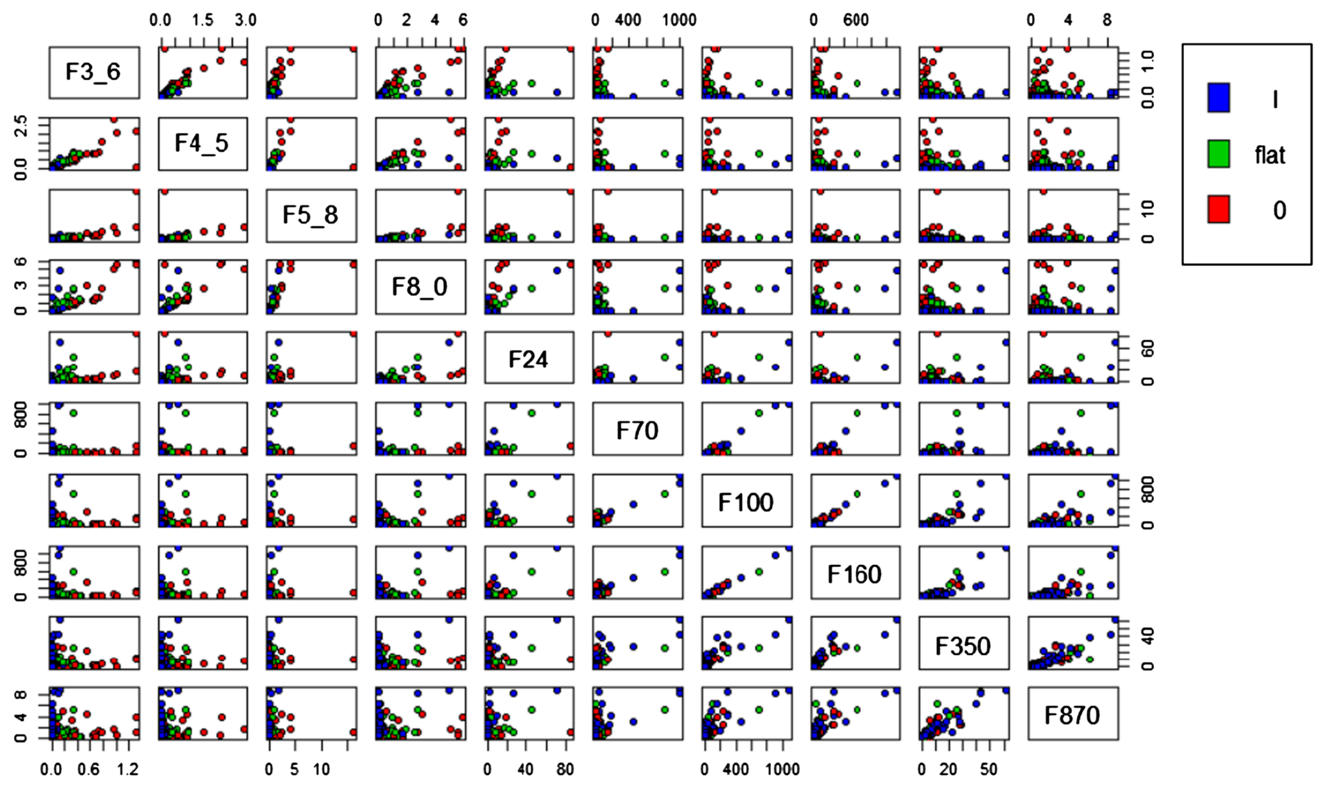


Fig. 2 Draftsman's plot showing the values of each of the considered continuum flux densities (in Jy) against each other

data cases (or rows). Hence, although the present data set is fairly small and hence subject to overfitting, both the k-fold CV and PCA employed in the present study can alleviate the influence of overfitting.

3.5 Machine learning classification

After data preparation, the following eight supervised classifiers were tested on the FFA16 data: a decision tree (e.g. Quinlan 1986; Murthy 1998), random forest (Ho 1995; Breinman 2001), gradient boosting machine (GBM; Breinman 1997; Friedman 2001), logistic regression (Cox 1958), naïve Bayes classifier (e.g. Domingos and Pazzani 1996; McCallum and Nigam 1998; Zhang 2004), k-nearest neighbours (k-NN; e.g. Cover and Hart 1967; Altman 1992), support vector machine (SVM; e.g. Vapnik and Lerner 1963; Cortes and Vapnik 1995; Burges 1998; Cristianini and Shawe-Taylor 2000; Scholkopf and Smola 2001), and artificial neural network (e.g. McCulloch and Pitts 1943; Rosenblatt 1958; Jeffrey and Rosner 1986; Zhang 2000). We refer to Kotsiantis et al. (2006b), Kotsiantis (2007), and Ball and Brunner (2010) for reviews of the aforementioned algorithms. As mentioned in Sect. 3.3, the classifiers were trained and their performance was tested using the technique of 10-fold CV. In what follows, is a brief description of each of the tested classifier.

3.5.1 Decision tree

A decision tree classifier attempts to learn simple decision rules that can predict the label for an instance on the basis of its feature values. The data are split according to the feature values in the so-called decision nodes, and the final leaf nodes contain the outputs (i.e. the protostellar classes in our case). One caveat of decision trees is that they are subject to overfitting if too complex trees are being built, and hence they might not generalise well.

To build a decision tree classifier, we used the `R` package `rpart` (Recursive Partitioning and Regression Trees), which uses the Classification and Regression Trees (CART; Breinman et al. 1984) algorithm. The CART algorithm employs binary splits on the input variables to grow the tree, and the splits were evaluated on the basis of the Gini index (a Gini score of zero means a perfect separation).

3.5.2 Random forest

Contrary to a single tree CART model, random forest takes random subsamples of both the observations and features from the training data (bagging), and trains decision trees on those cases. A whole army of such decision trees are grown, and the most common outcome for each observation is used as the final result. Such approach enable random forests to limit overfitting, which makes them very powerful classifiers.

For the purpose of random forest classification, we employed the `R` package `randomForest`, which implements Breinman's random forest algorithm. As the number of trees, we used the `randomForest`'s default value of $n_{\rm tree} = 500$, and the subsamples were chosen randomly with replacement. The number of variables that were randomly sampled was set at $\sqrt{p}$, where p is the total number of variables in the original data set ($p = 10$ in our case). The winning class was defined as the one with the highest ratio of proportion of votes to the cutoff parameter, where the latter was set at $1/k$, where k is the number of classes ($k = 3$ in our case). The minimum terminal node size, which controls the depth of the tree (the larger the parameter is, the smaller the tree will be), was set at unity, while in terms of the number of terminal nodes, the trees were let to grow to the maximum possible size.

3.5.3 Gradient boosting

While random forest is a bagging method with trees being run in parallel and without interaction, gradient boosting is an ensemble method where decision trees are added to learn the misclassification errors in existing models, and this sequentially boosts the training procedure. Because gradient boosting is a greedy algorithm (i.e. it makes the optimal choice at each step (locally optimal choice) as it tries to reach the overall optimal way to solve the classification problem), it can quickly overfit the training data set.

The boosting was performed using the `R`'s `gbm` algorithm, whose implementation follows the Friedman's GBM (Friedman 2001). Because our classification problem is composed of more than two classes, the analysis was carried out as a multinomial version. As the metric of the information retrieval measure, we used the normalised discounted cumulative gain (`metric = ndcg`). The total number of trees to fit, which corresponds to the number of iterations, was set at $n_{\rm tree} = 500$ as in the case of our random forest classification. The value of $n_{\rm tree}$ also corresponds to the number of basis functions that are being iteratively added in the boosting process (each additional basis function further reduces the loss function). The interaction depth, which represents the maximum depth of variable interactions, was fixed at $k = 3$. Hence, the number of terminal nodes or leaves, which is given by $J = k + 1$, was set at $J = 4$ (for comparison, $J = 2$, or a decision stump, means that no interactions between variables is allowed). The shrinkage parameter ($0 < \nu \leq 1$), which represent the learning rate, was set at $\nu = 0.005$. High learning rates of $\nu \simeq 1$ ($\nu = 1$ means no shrinking) are expected to result in overfit models, while small shrinkage parameter values ($\nu \leq 0.1$) slow down the learning process, but are expected to lead to much lower generalisation error (Friedman 1999). Finally, the minimum number of observations in the trees' terminal nodes was set to unity. Such small value was chosen because our training samples were so small.

3.5.4 Logistic regression

Despite its name, logistic regression is a classification algorithm rather than a regression technique. Logistic regression uses a logistic function and the predictor feature values to model the probabilities for an instance to belong to different classes.

To estimate a multinomial logistic regression model, we used the `R` algorithm called `multinom`. The algorithm predicted the probabilities for each source to belong to the three different classes (Class 0, Class I, and flat-spectrum sources), and the final assignment was done according to the highest predicted probability.

3.5.5 Naïve Bayes

Similarly to logistic regression, naïve Bayes classifiers belong to a family of probabilistic classifiers. Here, the classification relies on the Bayes' Theorem under the naïve assumption that the features are independent of each other.

To calculate the conditional posterior probabilities for our categorical protostellar class variable given the flux densities as predictor variables, we used the `naiveBayes` algorithm of `R`. No Laplace smoothing was applied, which would prevent the frequency-based probability estimate to be equal to zero as a result of a given class and feature value never occurring together in the training data. The latter is not an issue for the present data set, because the missing feature values were imputed.

3.5.6 *k*-Nearest neighbours

The k-NN algorithm is a member of the so-called instance-based, lazy-learning algorithms (Mitchell 1997). Here, a new, unseen instance is classified by comparing it with those k training cases that are closest in feature space (i.e. have similar properties with the new case), and the new case's class is then determined by a majority vote of its neighbours' classes. To find these k-nearest neighbours, the value of k needs to be specified, and a distance metric is required.

To carry out a k-NN classification, we used the `R`'s `knn3` algorithm. We run our k-NN classification by experimenting with different values of k, ranging from $k = 1$ to $k = 15$, and by adopting the Euclidean distance metric. As shown in Fig. 3, the best classification performance was reached when $k = 1$ (considering only the closest neighbour). The y-axis of Fig. 3 shows the overall accuracy of the classification,

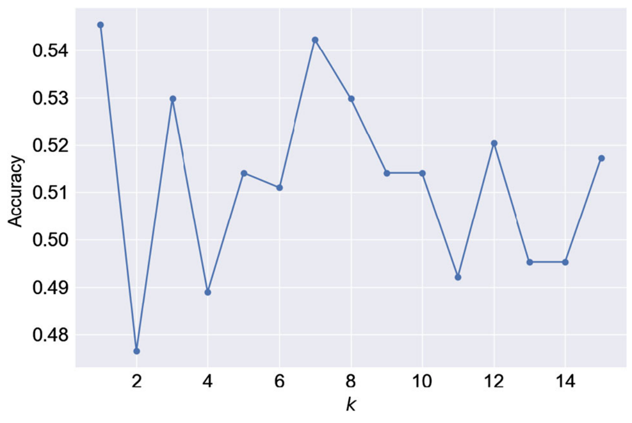


Fig. 3 Accuracy of the k-NN classifier as a function of the number of nearest neighbours. The best performance was reached when $k = 1$, but we selected the next best performance reached with $k = 7$ because the 1-NN classifier is subject to overfitting (see text for details)

which is defined as the ratio of cases that are correctly classified to the total number of cases. However, a 1-NN classifier can lead to overfitting and does not generalise well enough to other YSO samples (a small k means that noise has a higher effect on the classification). Hence, we consider the next highest accuracy that was reached when $k = 7$ in our subsequent comparison of different classifiers.

3.5.7 Support vector machine

The basic SVMs are binary classifiers, where the idea is to find the dividing hyperplane that both separates the two classes in the training data set and maximises the margin between the boundary members, that is between the SVs. A new instance is classified by examining on which side of the hyperplane it falls. In the case of non-linear classification, the goal is to find a hyperplane that is a non-linear function of the input variables. This is done by the so-called kernel trick, where the input features are mapped into a higher dimensional feature space. Besides binary classification, SVMs can also perform multiclass classification, which is the case in the present study ($k = 3$ classes). There are various options to do that, and we used a balanced one-against-one classification strategy, where three binary classifiers were trained (the number of classifiers is given by $k(k-1)/2$), and a simple voting strategy among them was applied to classify a new instance.

Our SVM classification was done using the `ksvm` algorithm of `R`, and to create a non-linear classifier we used a Gaussian radial basis function (RBF) kernel. The algorithm was set to calculate the inverse kernel width for the RBF directly from the data (rather than specifying its value). The value of the regularisation constant C was set equal to unity, where the effect of C is such that the larger it is, the narrower the margin between the SVs is, and hence the classifier is more prone to overfitting. A smaller C means a wider margin, and hence more misclassifications in the training set, which in turn can lead to underfitting issues.

3.5.8 Neural network

A neural network classifier reads in the input features in the so-called input layer, and, in the case of a multi-layer perceptron, attempts to learn a non-linear function approximator to correctly classify the training cases' target variables appearing in the so-called output layer.

To build a neural network classifier, we used the `R` package `nnet`, which fits a single-hidden-layer neural network, that is there is only one hidden layer between the aforementioned input and output layers. The feature data are being weighted, and transfered to the nodes or neurons in the hidden layer. The hidden layer neurons process the sum of the weighted inputs by applying a so-called transfer function, and pass the results forward. In the present work, we adopted a sigmoid shaped logistic transfer function, which is appropriate for discrete outputs such as protostellar classes. The maximum number of iterations used was set to 100, and the weight decay in the weight update rule was set equal to zero, which means that the weights did not exponentially decay to zero in case of no other updates were being scheduled (during the training phase, the update steps modify the weights applied on the input features).

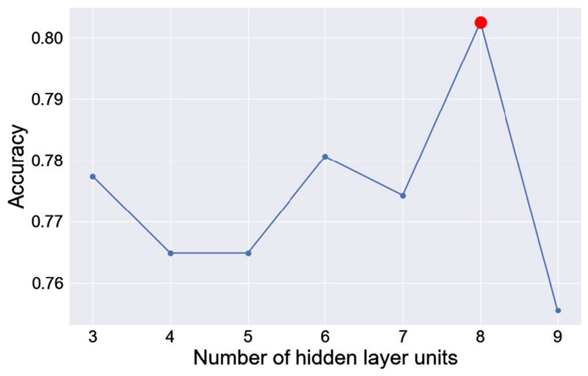
We experimented with different numbers of neurons in the hidden layer, ranging from two to nine, where the latter number is equal to the number of features (ten flux densities) minus one. Using too many layers or neurons in the net can lead to overfitting, and hence we only employed a single-hidden-layer model. As shown in the left panel in Fig. 4, the best performance was found when there were eight neurons. The corresponding neural net is shown in the right panel in Fig. 4.

4 Results and discussion

4.1 Performance metrics

To quantify and visualise the performance of each of the aforementioned classification algorithm, we derived their confusion matrices (see Fig. 5). The columns of each matrix represent the instances in an actual, SED-based class (FFA16), while the rows represent the instances in a predicted class. The diagonal elements of a confusion matrix show the cases where the predicted class is the same as the true (SED-based) class, while the off-diagonal elements represent the misclassified cases. Hence, the larger the diagonal element numbers are, the better the classifier has performed.

Several different parameters that characterise the performance of a classifier can be calculated from the confusion



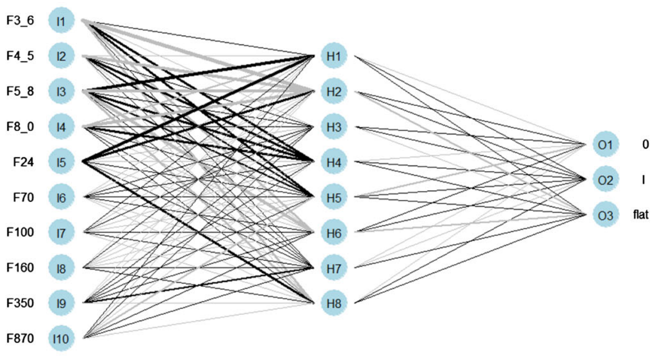


Fig. 4 *Left:* Accuracy of the single-layer neural network classifier as a function of the number of neurons or nodes in the hidden layer. The red, filled circles indicates the highest accuracy, which was obtained with eight nodes. *Right:* Artificial neural net comprised of eight hidden layer nodes (labelled as H1 through H8), which was found to yield the best accuracy among the tested neural nets as shown in the left panel. The input features (labelled as I1–I10) are the ten continuum flux densities from FFA16, and the outputs are the protostellar classes (Class 0, Class I, and flat-spectrum sources, labelled as O1, O2, and O3, respectively). The black lines between the layers represent positive weights, while the grey lines indicate negative weights. The thickness of the lines is proportional to the magnitude of the weight with respect to all other weights

matrix (e.g. Fawcett 2006), but here we focus only on four of them, namely the aforementioned overall accuracy, which tells how often the classifier is correct (the sum of the diagonal elements of the confusion matrix divided by the total number of cases), purity of a class (ratio between the correctly classified sources of a class and the number of sources classified in that class), contamination of a class (ratio between the misclassified sources in a class and the number of sources classified in that class), which is given by contamination = 1 − purity (e.g. Tangaro et al. 2015), and the Matthews correlation coefficient (Matthews 1975), or the phi coefficient, which is defined as

$$\mathrm{MCC} = \frac{\mathrm{TP}\times\mathrm{TN}-\mathrm{FP}\times\mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FP})\times(\mathrm{TP}+\mathrm{FN})\times(\mathrm{TN}+\mathrm{FP})\times(\mathrm{TN}+\mathrm{FN})}}, \tag{1}$$

where TP, TN, FP, and FN are the numbers of true positives, true negatives, false positives, and false negatives, respectively. The MCC can be considered a correlation coefficient between the true and predicted binary classifications, and its value lies in the range of -1 to $+1$, where -1 means a full disagreement between the predicted and true classes, 0 is equivalent to random guess, and $+1$ indicates a perfect prediction performance. Because we are dealing with a multiclass classification (rather than binary classification), we calculated the so-called micro-averaged MCCs, that is we summed all the TP, TN, FP, and FN values for each class to calculate the MCC. More precisely, the TPs were derived by taking the sum of the confusion matrix diagonal elements, the TNs were calculated by removing the target class' row and column from the confusion matrix, and then taking the sum of all the remaining elements, the FPs were calculated as the sum of the respective column, minus the diagonal element for the class under consideration, and the FNs were computed by taking the sum of the respective row elements, minus the diagonal elements. The performance metrics are tabulated in Table 1.

4.2 Performance of the tested classifiers

The evolutionary stages of our target protostellar objects were originally derived by FFA16 using an SED analysis. Hence, by comparing our classifications with respect to these SED-based classes, we are assuming that the SED classes are correct. Although panchromatic SEDs are expected to yield some of the most reliable source evolutionary stages (if not even the most reliable ones), it is still good to keep in mind the possibility that a machine learning classifier could predict a correct evolutionary stage for a source although it would differ from its SED class. The SED-based source classification itself can depend on the exact method of how the analysis is performed (e.g. modified blackbody fitting versus fitting based on radiative transfer models as done in FFA16). Moreover, the assumptions about the dust grain properties affect the dust-based physical properties of the source, and hence the corresponding evolutionary stage. From an observational point of view, the source inclination angle and variability might also affect the inferred evolutionary stage (e.g. the central protostar might be visible if observed through the outflow cavity). Related to the issues of the SED analysis, we remind the reader that all the FFA16 sources were assumed to lie at the same distance (Sect. 2). Hence, we did not use the distance as a separate feature in our supervised source classification. However, if the source sample is being drawn from different star-forming regions that lie at different distances from the Sun, the distance should be included as a feature because some of the fundamental source properties depend on it (e.g. the mass scales as d^2).

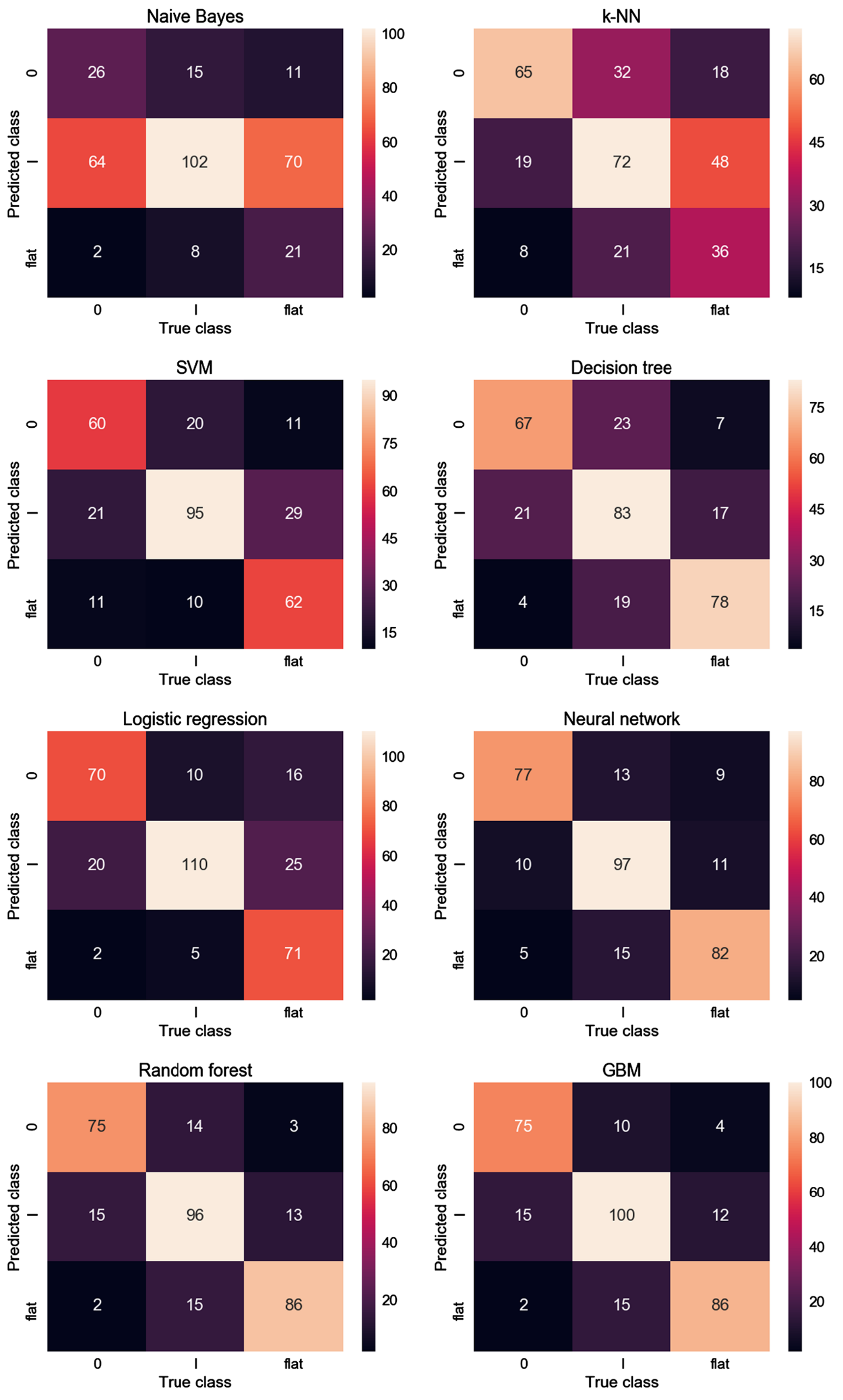


Fig. 5 Colour-coded confusion matrices showing the performance of the tested machine learning classifiers

Table 1 Performance metrics of the tested machine learning classifiers in order of increasing accuracy

Classifier	Accuracy	Purity[a]	Contamination[a]	MCC
Naïve Bayes	0.47	0.50, 0.69, 0.68	0.50, 0.53, 0.32	0.20
k-NN	0.54[b]	0.57, 0.52, 0.55	0.43, 0.48, 0.45	0.31
SVM	0.68	0.66, 0.66, 0.75	0.34, 0.34, 0.25	0.52
Decision tree	0.71	0.69, 0.17, 0.77	0.31, 0.83, 0.23	0.57
Logistic regression	0.79	0.81, 0.71, 0.91	0.19, 0.29, 0.09	0.68
Neural network	0.80[c]	0.78, 0.82, 0.80	0.22, 0.18, 0.20	0.70
Random forest	0.81	0.82, 0.77, 0.85	0.18, 0.23, 0.15	0.71
GBM	0.82	0.84, 0.79, 0.83	0.16, 0.21, 0.17	0.73

[a]The three values reported for the purity and contamination refer to the Class 0, Class I, and flat-spectrum sources

[b]This accuracy was reached using a value of $k = 7$ (Fig. 3). With $k = 1$, the accuracy was about 0.55, but such model is likely subject to overfitting (Sect. 3.5.6)

[c]This accuracy was reached with eight neurons in a single-hidden-layer neural net (see Fig. 4)

In the following subsections, we briefly discuss the performance of each tested classifier. The algorithms are discussed in the order of increasing classification accuracy, which were found to range from 47% to 82% (Table 1).

4.2.1 Naïve Bayes

The poorest job in the present classification analysis was done by the naïve Bayes classifier with an accuracy of only 47% and an MCC of 0.20. For comparison, among three possible classes as in the present study (Class 0, Class I, and flat sources), the accuracy of random guess would be $\sim 33\%$. As described in Sect. 3.5.5, naïve Bayes classifier is based on the assumption that the predictors are independent of each other. Considering the present set of features, which is composed of continuum flux densities, the assumption of their independence is certainly violated. For example, as mentioned in Sect. 3.2, the far-IR and submm flux densities explored in the present work depend on each other via dust opacity.

4.2.2 k-Nearest neighbours

With an accuracy of only 54% and an MCC of 0.31, the k-NN classifier was found to be comparable to the naïve Bayes classifier. As described in Sect. 3.5.6, the number of nearest neighbours we considered was set to $k = 7$. The decision of how many neighbours to take into account controls the model's ability to generalise to future data instances. Although the exact value of the optimal k is dependent on the analysed data set, there are a few general things to keep in mind. First, if only a single nearest neighbour is considered, the classifier is subject to noisy data features. For this reason, we did not adopt the value $k = 1$ although it yielded a better accuracy than using $k = 7$. Secondly, while a large k reduces the influence of noisy data, it is computationally more expensive and suffers from the possibility of ignoring important, small-scale patterns (and one might end up considering cases that are not even actual neighbours anymore). Hence, the optimal k is expected to lie somewhere between these two extreme cases (e.g. Lantz 2015). A common empirical rule of thumb is to set k equal to $\sqrt{n_{\rm train}}$, where $n_{\rm train}$ is the size of the training data set (e.g. Hassanat et al. 2014, and references therein; Lantz 2015). This will usually lead to large values of k (i.e. many neighbours), which reduces the effect of variance caused by noisy data. In our 10-fold CV analysis, nine folds were used to train the k-NN classifier, which means that $n_{\rm train}$ was roughly ~ 270, which would suggest a value of $k \simeq 16$. The optimal number of nearest neighbours we found (in terms of accuracy), $k = 7$, is over two times smaller than suggested by the aforementioned rule of thumb. Hence, although possibly being time-consuming, the best value of k should probably be searched using a cross-validation approach as in the present study (see Fig. 3).

4.2.3 Support vector machine

The accuracy of our SVM classifier, 68%, is a factor of 1.26 better than that of our k-NN classifier. Moreover, the MCC of our SVM (MCC = 0.52) just exceeds a binary classification threshold of 0.50 between pure guessing (MCC = 0) and perfect prediction (MCC = 1). By tuning the hyperparameters of the SVM, such as the C parameter, the classification accuracy could potentially be improved, although the risk for overfitting might increase accordingly.

4.2.4 Decision tree

A simple decision tree algorithm was found to perform fairly well as compared to the other algorithms tested in the present work. The accuracy and MCC (0.71 and 0.57) of

Table 2 Classification accuracy when one of the ten continuum bands was left out of consideration

Classifier	Wavelength band [µm] ignored									
	3.6[a]	4.5[a]	5.8[a]	8.0[a]	24[b]	70[c]	100[c]	160[c]	350[d]	870[d]
Naïve Bayes	0.45	0.46	0.47	0.47	0.49	0.46	0.47	0.47	0.46	0.46
k-NN[e]	0.54	0.54	0.54	0.53	0.51	0.54	0.53	0.61	0.54	0.53
SVM	0.66	0.68	0.68	0.68	0.68	0.68	0.68	0.70	0.65	0.68
Decision tree	0.70	0.71	0.72	0.71	0.74	0.72	0.73	0.71	0.71	0.71
Logistic regression	0.75	0.77	0.77	0.77	0.73	0.75	0.75	0.77	0.77	0.77
Neural network[f]	0.76	0.77	0.73	0.78	0.70	0.74	0.77	0.76	0.79	0.76
Random forest	0.79	0.81	0.82	0.81	0.76	0.81	0.82	0.82	0.82	0.81
GBM	0.76	0.82	0.81	0.82	0.77	0.79	0.82	0.81	0.81	0.82

[a]*Spitzer*/IRAC band

[b]*Spitzer*/MIPS band

[c]*Herschel*/PACS band

[d]APEX bolometer band

[e]The number of nearest neighbours was fixed at $k = 7$

[f]The number of nodes in the hidden layer was fixed at eight

our decision tree classifier are comparable to those derived for the SVM.

4.2.5 Logistic regression

Our multinomial logistic regression yielded a classification accuracy of 79% with fairly pure classes (purity is 0.81, 0.71, and 0.91 for the Class 0, Class I, and flat-spectrum sources, respectively). The derived MCC of 0.68 is a factor of 1.19 larger than for our decision tree model.

4.2.6 Neural network

Eighty percent of the test cases were correctly classified by our neural network classifier. Also, an MCC of 0.70 derived for the classifier shows that its prediction performance is fairly good among the tested algorithms. Overall, the performance metrics of our neural network are comparable to our logistic regression. This is perhaps unsurprising, because the transfer function in our neural net was taken to be the logistic sigmoid function.

4.2.7 Random forest

The second highest classification accuracy (81%) and an MCC (0.71) were derived for our random forest classifier. As expected, a random forest classifier did a much better job than a simple decision tree, which is an indication that the former generalises better on unseen data than the latter.

4.2.8 Gradient boosting

The best classification performance (82% accuracy) was obtained with a GBM. Our gradient boosting classifier led to fairly pure classifications per class (0.84, 0.79, and 0.83 for the Class 0, Class I, and flat-spectrum sources), and hence low contaminations. Its MCC of 0.73 also indicates a reasonable prediction performance. As expected, GBM outperforms the simple decision tree (a factor of 1.15 better accuracy), while our GBM was found to be only marginally (factor of 1.01) more accurate than our random forest classifier with an accuracy of 81% and MCC of 0.71 (the second most accurate classifier in the present study).

4.3 Unveiling the most important wavelength bands for classifying young stellar objects via supervised learning algorithms

The PCA presented in Sect. 3.4 suggests that the 3.6 µm flux density is the most informative feature in the preset study (the feature explains 56.1% of the variance of the data; see Fig. 1). For comparison, the next most important band was found to be the 4.5 µm band, which explains 18.1% of the data variance. As an alternative approach to unveil the most important band for the present YSO classification, we employed the leave-one-out cross-validation (LOOCV) technique, that is the classifications were done by using nine out of the ten features (ranging from the *Spitzer*/IRAC 3.6 µm band to LABOCA 870 µm band), and this process was repeated ten times with a different wavelength band left out every time.

In Table 2, we tabulate the classification accuracies of each of the tested algorithm when one out of the ten most relevant features was being left out. The results are presented visually in Fig. 6. We note that in the cases of the *k*-NN and neural network classifiers, the hyperparameter settings (the k value and number of neurons) were identical to those used for the full feature set. The results suggest that if the *Spitzer*/IRAC 3.6 µm band is ignored, the classification accuracies drop with respect to the full feature set (though only by factors of 1–1.08), which conforms with the PCA result of 3.6 µm band being the most informative one. The importance of the 3.6 µm band could, at least partly, be related to the shocked H_2 emission associated with protostellar outflows, although such shock emission is stronger at 4.5 µm (e.g. Ybarra and Lada 2009). There is also a polycyclic aromatic hydrocarbon feature at 3.3 µm owing to C-H stretching that might contribute to the *Spitzer*/IRAC band 1 emission (e.g. Draine 2003 for a review). Also, the *Spitzer*/MIPS 24 µm band, which is sensitive to warm dust emission (e.g. Rathborne et al. 2010), appears to be a fairly important feature for most of the classifiers; if the band is ignored, the classification accuracies drop by factors of 0.96–1.14. For the decision tree and naïve Bayes classifiers, however, the classification accuracy was actually marginally higher when the 24 µm data were ignored (by a factor of 1.04 in both cases). From a physical point of view, the most important wavelength bands are expected to be those probing the peak of the source SED (i.e. around $\sim$ 100 µm; see Sect. 2), but this is not manifested in our LOOCV feature selection.

As mentioned above, in some cases it was found that the exclusion of a wavelength band actually increases the classification accuracy with respect to the case where all the ten features are being used. Most notably, this happens when the *Herschel*/PACS 160 µm band is ignored from the k-NN classification (the accuracy increases by a factor of 1.13; see the green curve in Fig. 6). This suggests that the inclusion of the aforementioned band might have led to a slight overfitting effect in our k-NN classification. We also note that the random forest and GBM are generally found to yield the best classification accuracies when ignoring the different features, but in the case where the 3.6 µm band was ignored, our neural network appeared equally good as the GBM (76% accuracy for both). In the latter case, the random forest was found to yield the highest classification accuracy of 79%, but the difference is only marginal with respect to the GBM. Also, these small differences in the accuracies compared to the usage of all the ten features can not be considered significant owing to the random sampling nature of the both the random forest algorithm and the 10-fold CV technique.

Of course, there are many feature combinations among the considered flux densities that could be explored. For example, there are 45 unique flux density pairs in the present set of ten different flux densities. However, a thorough feature analysis is beyond the scope of the present study.

Considering the future YSO surveys where similar machine learning approaches could be used as in the present study, the observed wavelengths are likely to differ from those we have analysed (e.g. the cryogenic phase of *Spitzer*

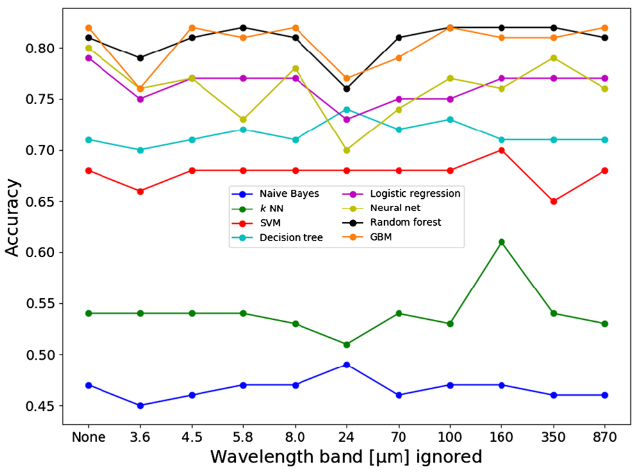


Fig. 6 Behaviour of the classification accuracy when one of the ten continuum bands was left out of consideration. For reference, the first data point from left shows the accuracy when all the ten features were used (see column (2) in Table 1)

operated from 2003 to 2009, while its warm mission (using the 3.6 µm and 4.5 µm IRAC bands only) is scheduled to end in March 2019 (e.g. Hora et al. 2012; Yee et al. 2017), and the *Herschel* mission ended on 29 April 2013 when the satellite ran out of its helium coolant (e.g. Sauvage et al. 2014)). Nevertheless, the aforementioned analysis suggests that bands near 3.6 µm and 24 µm would be informative if the other bands are comparable to those employed here.

4.4 Potential of using machine learning in classifying young stellar objects

Owing to the fact that we considered only ten out of the original 29 features (i.e. the 2MASS and *Spitzer*/IRS flux densities were ignored), a protostellar classification accuracy of 82% with a GBM can be considered fairly good. Also, although being the largest, homogeneous protostar sample drawn from a single molecular cloud system, the size of the employed data set is fairly small for a machine learning approach (only 319 sources in total), and undoubtedly a higher classification accuracy could be reached with a larger training sample. On the other hand, as shown in Fig. 2, the Orion Class 0, Class I, and flat-spectrum sources from FFA16 are not well separable in the two-dimensional projections of the feature space, which is an indication that learning their classification is demanding for at least some of the supervised classifiers.

A more detailed missing value imputation, such as estimating the missing far-IR to submm flux densities in a source-by-source fashion from the existing values by assuming a value of β (see Sect. 3.2), could lead to an improved performance. However, this approach is also based on assumptions about the flux frequency dependence at different bands. Finally, an application of more advanced ensemble methods, like the extreme gradient boosting (XGBoost;

Chen and Guestrin 2016), has the potential to lead to an improved accuracy.

As mentioned in Sect. 4.3, the future YSO surveys where machine learning assisted source classification could be used will undoubtedly be carried out at least partly at different wavelengths than considered here. This also means that the models trained in the present work can not be employed as such, but they would need to be retrained (cf. the deep learning knowledge transfer study by Domínguez Sánchez et al. 2018). In fact, even in a hypothetical case where there would be a survey of YSOs carried out at exactly the same ten bands as we considered, the rms noise levels of the observations would probably still be different, which would again call for retraining the classifiers (for example, deeper surveys could detected weaker sources than those in the FFA16 Orion sample). Regarding this issue, we note that the 10σ limiting magnitudes in the *Spitzer*/IRAC 3.6, 4.5, 5.8, and 8 µm data, and the *Spitzer*/MIPS 24 µm data employed by FFA16 were 16.5, 16.0, 14.0, 13.0, and 8.5 mag, respectively (Megeath et al. 2012; Spezzi et al. 2015), and Megeath et al. (2012) derived the final magnitudes for all their *Spitzer* sources that were detected with uncertainties of ≤ 0.25 mag in one of the four IRAC bands. The properties of the *Herschel* and APEX data products employed by FFA16 will be described in more detail by B. Ali et al. (in prep.) and T. Stanke et al. (in prep.); see Fischer et al. (2017). Moreover, the source flux densities depend on the aperture sizes used to extract the photometry, and if the apertures differ from those used by FFA16 (who, for example, used the aperture radii of $9''.6$, $9''.6$, and $12''.8$ for the *Herschel* 70, 100, and 160 µm data, respectively), the present classifiers would again have to be retrained.

Although reaching high accuracies, a supervised machine learning classification cannot replace an SED-based YSO classification because an SED analysis also yields the important physical properties of the source, like the dust temperature and (envelope) dust mass (however, the SED fitting itself could also rely on machine learning regression). Moreover, SED analyses are still expected to be needed to create the training data sets for machine learning applications. Nevertheless, if appropriate training data sets are available, machine learning techniques can serve as a quick way to estimate the relative percentages of YSOs in different evolutionary stages in the era of big astronomical data (e.g. Hocking et al. 2018; Pearson et al. 2018), and also to mine the YSO data to unveil interesting subsamples for more detailed follow-up observations (e.g. Marton et al. 2016).

5 Summary and conclusions

We used eight different supervised machine learning algorithms to classify the Orion protostellar objects from FFA16 into Class 0, Class I, and flat-spectrum sources. On the basis of PCA, we employed only the IR and submm continuum photometric data from FFA16. The training and testing of the classifiers were performed by using a 10-fold CV technique. Using the SED-based classifications of FFA16 as the benchmark, we found that the highest classification accuracy is reached by a GBM algorithm (82% of the cases were correctly classified with $\gtrsim 80\%$ purity and an MCC of 0.73), while the poorest performance was that of naïve Bayes classification (47% accuracy).

Our analysis suggests that among the ten continuum emission bands used in the classification, the *Spitzer* 3.6 µm and 24 µm flux densities are the most informative features in terms of the source classification accuracy. Hence, these two wavelength bands would be useful to include in a panchromatic YSO classification study, especially if the other bands available are comparable to those analysed in the present work (i.e. 4.5, 5.8, 8.0, 70, 100, 160, 350, and 870 µm).

Larger data sets, detailed missing value imputations, and more sophisticated learning algorithms have the potential to improve the classification accuracies. Overall, machine learning algorithms can provide a fast (at least compared to an SED analysis) way to classify large samples of sources into different evolutionary stages, and hence estimate the statistical lifetimes of the sources, and also pick up subsamples of interesting sources for targeted follow-up studies. However, an obvious challenge of supervised machine learning classification is the creation of training data sets, which requires classification of large numbers of YSOs into different evolutionary stages on the basis of their measured flux densities at the observed wavelengths. Because the latter is based on SED fitting, which itself requires knowledge of the source distance and assumptions about the underlying dust grain model, an SED analysis might be a prerequisite to the usage of machine learning classifiers on new survey data sets.

Acknowledgements I would like to thank the referee for providing constructive comments and suggestions that helped to improve the quality of this paper. This research has made use of NASA's Astrophysics Data System and the NASA/IPAC Infrared Science Archive, which is operated by the JPL, California Institute of Technology, under contract with the NASA.

References

Abdi, H., Williams, L.J.: Wiley Interdiscip. Rev.: Comput. Stat. **2**(4), 433–459 (2010)
Adams, F.C., Lada, C.J., Shu, F.H.: Astrophys. J. **312**, 788 (1987)
Alpaydin, E.: Introduction to Machine Learning, 2nd edn. MIT Press, Cambridge (2010)
Altman, N.S.: Am. Stat. **46**(3), 175–185 (1992)
An, F., Stach, S.M., Smail, I., et al.: Astrophys. J. **862**(2), 101 (2018). arXiv:1806.06859
André, P., Montmerle, T.: Astrophys. J. **420**, 837 (1994)

André, P., Ward-Thompson, D., Barsony, M.: Astrophys. J. **406**, 122 (1993)
André, P., Ward-Thompson, D., Barsony, M.: In: Mannings, V., Boss, A.P., Russell, S.S. (eds.) Protostars and Planets IV, p. 59. University of Arizona Press, Tuscon (2000)
André, P., Men'shchikov, A., Bontemps, S., et al.: Astron. Astrophys. **518**, L102 (2010)
Aniyan, A.K., Thorat, K.: Astrophys. J. Suppl. Ser. **230**, 20 (2017)
Ball, N.M., Brunner, R.J.: Int. J. Mod. Phys. D **19**, 1049 (2010)
Beck, M.R., Scarlata, C., Fortson, L.F., et al.: Mon. Not. R. Astron. Soc. **476**(4), 5516–5534 (2018). arXiv:1802.08713
Box, G.E.P., Meyer, R.D.: Technometrics **28**(1), 11–18 (1986)
Breinman, L.: Technical Report 486, Statistics Department, University of California, Berkeley, CA 94720 (1997)
Breinman, L.: Mach. Learn. **45**(1), 5–32 (2001)
Breinman, L., Friedman, J.H., Stone, C.J., Olshen, R.A.: Classification and Regression Trees. Taylor & Francis, London (1984)
Burges, C.: Data Min. Knowl. Discov. **2**(2), 1–47 (1998)
Cawley, G.C., Talbot, N.L.C.: J. Mach. Learn. Res. **11**, 2079–2107 (2010)
Chen, T., Guestrin, C.: arXiv:1603.02754 (2016)
Cortes, C., Vapnik, V.N.: Mach. Learn. **20**(3), 273–297 (1995)
Cover, T., Hart, P.: IEEE Trans. Inf. Theory **13**(1), 21–27 (1967)
Cox, D.R.: J. R. Stat. Soc. B **20**, 215–242 (1958)
Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press, Cambridge (2000)
Domingos, P., Pazzani, M.: In: Saitta, L. (ed.) Proceedings of the Thirteenth International Conference on Machine Learning, pp. 105–112. Morgan Kaufmann, San Francisco (1996)
Domínguez Sánchez, H., Huertas-Company, M., Bernardi, M., et al.: Mon. Not. R. Astron. Soc. (2018, in press). arXiv:1807.00807
Draine, B.T.: Annu. Rev. Astron. Astrophys. **41**, 241 (2003)
Dunham, M.M., Stutz, A.M., Allen, L.E., et al.: In: Beuther, H., Klessen, R.S., Dullemond, C.P., Henning, Th. (eds.) Protostars and Planets VI, p. 195. University of Arizona Press, Tucson (2014). 914 pp.
Dunham, M.M., Allen, L.E., Evans, N.J. II, et al.: Astrophys. J. Suppl. Ser. **220**, 11 (2015)
Evans, N.J. II, Dunham, M.M., Jørgensen, J.K., et al.: Astrophys. J. Suppl. Ser. **181**, 321 (2009)
Fawcett, T.: Pattern Recognit. Lett. **27**(8), 861–874 (2006)
Fazio, G.G., Hora, J.L., Allen, L.E., et al.: Astrophys. J. Suppl. Ser. **154**, 10 (2004)
Fischer, W.J., Megeath, S.T., Furlan, E., et al.: Astrophys. J. **840**, 69 (2017)
Friedman, J.H.: Comput. Stat. Data Anal. **38**, 367–378 (1999)
Friedman, J.H.: Ann. Stat. **29**(5), 1189–1232 (2001)
Furlan, E., Fischer, W.J., Ali, B., et al.: Astrophys. J. Suppl. Ser. **224**, 5 (2016). FFA16
Greene, T.P., Wilking, B.A., André, P., et al.: Astrophys. J. **434**, 614 (1994)
Güsten, R., Nyman, L.Å., Schilke, P., et al.: Astron. Astrophys. **454**, L13 (2006)
Hassanat, A.B., Mohammad, A.A., Altarawneh, G.A., et al.: Int. J. Comput. Sci. Inf. Secur. **12**(8), 33–39 (2014)
Hawkins, D.M.: J. Chem. Inf. Comput. Sci. **44**(1), 1–12 (2004)
He, H., Ma, Y.: Imbalanced Learning: Foundations, Algorithms, and Applications, 1st edn. Wiley–IEEE Press, New York (2013)
Ho, T.K.: In: Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995, pp. 278–282 (1995)
Hocking, A., Geach, J.E., Sun, Y., Davey, N.: Mon. Not. R. Astron. Soc. **473**, 1108 (2018)
Hora, J.L., Marengo, M., Park, R., et al.: Proc. SPIE **8442**, 844239 (2012)
Hotelling, H.: J. Educ. Psychol. **24**(6), 417 (1933)
Houck, J.R., Roellig, T.L., van Cleve, J., et al.: Astrophys. J. Suppl. Ser. **154**, 18 (2004)
Hui, J., Aragon, M., Cui, X., Flegal, J.M.: Mon. Not. R. Astron. Soc. **475**, 4494 (2018)
James, G., Witten, D., Hastie, T., Tibshirani, R.: An Introduction to Statistical Learning with Applications in R, 8th edn. Springer, New York (2017)
Jeffrey, W., Rosner, R.: Astrophys. J. **310**, 473 (1986)
Jolliffe, I.: Principal Component Analysis. Springer, Berlin (2002)
Kotsiantis, S.B.: Informatica **31**, 249–268 (2007)
Kotsiantis, S.B., Kanellopoulos, D., Pintelas, P.E.: GESTS Int. Trans. Comput. Sci. Eng. **30**(1), 26–36 (2006a)
Kotsiantis, S.B., Zaharakis, I.D., Pintelas, P.E.: Artif. Intell. Rev. **26**, 159–190 (2006b)
Krakowski, T., Małek, K., Bilicki, M., et al.: Astron. Astrophys. **596**, A39 (2016)
Lada, C.J.: In: IAU Symposium 115: Star Forming Regions, pp. 1–18. Reidel, Dordrecht (1987)
Lada, C.J., Wilking, B.A.: Astrophys. J. **287**, 610 (1984)
Lantz, B.: Machine Learning with R, 2nd edn. Packt, Birmingham (2015)
Little, R.J.A.: Missing data adjustments in large surveys (with discussion). J. Bus. Econ. Stat. **6**, 287–301 (1988)
Lochner, M., McEwen, J.D., Peiris, H.V., et al.: Astrophys. J. Suppl. Ser. **225**, 31 (2016)
Lukic, V., Brüggen, M., Banfield, J.K., et al.: Mon. Not. R. Astron. Soc. **476**, 246 (2018)
Marton, G., Tóth, L.V., Paladini, R., et al.: Mon. Not. R. Astron. Soc. **458**, 3479 (2016)
Matthews, B.W.: Biochim. Biophys. Acta, Protein Struct. **405**(2), 442–451 (1975)
McCallum, A., Nigam, K.: In: AAAI-98 Workshop on Learning for Text Categorization, vol. 752, pp. 41–48 (1998)
McCulloch, W., Pitts, W.H. Jr.: Bull. Math. Biophys. **5**(4), 115–133 (1943)
Megeath, S.T., Gutermuth, R., Muzerolle, J., et al.: Astron. J. **144**, 192 (2012)
Miettinen, O.: Astrophys. Space Sci. **361**, 248 (2016)
Miettinen, O., Harju, J., Haikala, L.K., et al.: Astron. Astrophys. **500**, 845 (2009)
Mitchell, T.: Machine Learning. McGraw-Hill, New York (1997)
Mosteller, F., Turkey, J.W.: Data analysis, including statistics. In: Lindzey, G., Aronson, E. (eds.) Handbook of Social Psychology, vol. 2. Addison-Wesley, Reading (1968)
Murthy, S.K.: Data Min. Knowl. Discov. **2**, 345–389 (1998)
Myers, P.C., Ladd, E.F.: Astrophys. J. Lett. **413**, L47 (1993)
Pashchenko, I.N., Sokolovsky, K.V., Gavras, P.: Mon. Not. R. Astron. Soc. **475**, 2326 (2018)
Pearson, K.: Philos. Mag. **2**(11), 559–572 (1901)
Pearson, K.A., Palafox, L., Griffith, C.A.: Mon. Not. R. Astron. Soc. **474**, 478 (2018)
Pilbratt, G.L., Riedinger, J.R., Passvogel, T., et al.: Astron. Astrophys. **518**, L1 (2010)
Poglitsch, A., Waelkens, C., Geis, N., et al.: Astron. Astrophys. **518**, L2 (2010)
Quinlan, J.R.: Induction of decision trees. Mach. Learn. **1**(1), 81–106 (1986)
Rathborne, J.M., Jackson, J.M., Chambers, E.T., et al.: Astrophys. J. **715**, 310 (2010)
Rieke, G.H., Young, E.T., Engelbracht, C.W., et al.: Astrophys. J. Suppl. Ser. **154**, 25 (2004)
Rosenblatt, F.: Psychol. Rev. **65**(6), 386–408 (1958)
Saar-Tsechansky, M., Provost, F.: J. Mach. Learn. Res. **8**, 1625–1657 (2007)

Sauvage, M., Okumura, K., Klaas, U., et al.: Exp. Astron. **37**, 397 (2014)

Scholkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge (2001)

Shetty, R., Kauffmann, J., Schnee, S., Goodman, A.A., Ercolano, B.: Astrophys. J. **696**, 2234 (2009)

Siringo, G., Kreysa, E., Kovács, A., et al.: Astron. Astrophys. **497**, 945 (2009)

Siringo, G., Kreysa, E., De Breuck, C., et al.: Messenger **139**, 20 (2010)

Skrutskie, M.F., Cutri, R.M., Stiening, R., et al.: Astron. J. **131**, 1163 (2006)

Spezzi, L., Petr-Gotzens, M.G., Alcalá, J.M., et al.: Astron. Astrophys. **581**, A140 (2015)

Sreejith, S., Pereverzyev, S. Jr., Kelvin, L.S., et al.: Mon. Not. R. Astron. Soc. **474**, 5232 (2018)

Stutz, A.M., Tobin, J.J., Stanke, T., et al.: Astrophys. J. **767**, 36 (2013)

Tangaro, S., Amoroso, N., Brescia, M., et al.: Comput. Math. Methods Med. **2015**, 814104 (2015)

van Buuren, S., Groothuis-Oudshoorn, K.: J. Stat. Softw. **45**(3), 1–67 (2011)

Vapnik, V., Lerner, A.: Autom. Remote Control **24**, 774–780 (1963)

White, R.J., Greene, T.P., Doppmann, G.W., et al.: In: Reipurth, B., Jewitt, D., Keil, K. (eds.) Protostars and Planets V, p. 117. University of Arizona Press, Tucson (2007). 951 pp.

Witten, I.H., Frank, E.: Data Mining—Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann/Elsevier Inc., San Mateo/Amsterdam (2005)

Yan, Q.-Z., Xu, Y., Walsh, A.J., et al.: Mon. Not. R. Astron. Soc. **476**, 3981 (2018)

Ybarra, J.E., Lada, E.A.: Astrophys. J. Lett. **695**, L120 (2009)

Yee, J.C., Fazio, G.G., Benjamin, R., et al.: arXiv:1710.04194 (2017)

Zhang, G.: IEEE Trans. Syst. Man Cybern., Part C, Appl. Rev. **30**(4), 451–462 (2000)

Zhang, H.: Astron. Astrophys. **1**(2), 3 (2004)