MAGIC GAMMA TELESCOPE
Machine Learning project:
Classifying particles from MAGIC observations
February 2022

Georgiadis Stefanos

Abstract. This study explores the use of machine learning to analyze a real data set of interaction products from gamma rays and particles in Earths atmosphere. [1]. Machine learning methods, such as logistic regression, random forests, decision trees, support vector machine, linear discriminant analysis and quadratic discriminant analysis were employed in order to determine the most important features required to draw conclusions, make predictions and finally make suggestions. Five of the features were found to have sufficiently high correlations with the target. Therefore, it was possible to train the data set and find out that using these five features, it is possible to discriminate statistically those caused by primary gammas (signal) from the images of hadronic showers initiated by cosmic rays in the upper atmosphere (background). [1].

Keywords: MAGIC, gamma-rays, Cherenkov, Decision Trees, Logistic Regression, Random Forests, Support Vector Machine, Linear Discriminant Analysis, Quadratic Discriminant Analysis

1 Introduction

The main purpose of this work is to employ the knowledge gained from the machine learning course and practice to a preferred data set of choice. The MAGIC gamma telescope dataset [1] was chosen for this study. Since its output is binary, it qualifies as a classification problem and relevant methods of analysis were selected to study its features and behaviour. The theoretical framework of each of the analysis tools employed are given in the following subsections. The methodology and results obtained from each analysis are then discussed separately in the next sections. In the end, a comparison is made among all the different methods used and a final conclusion is derived from the study.

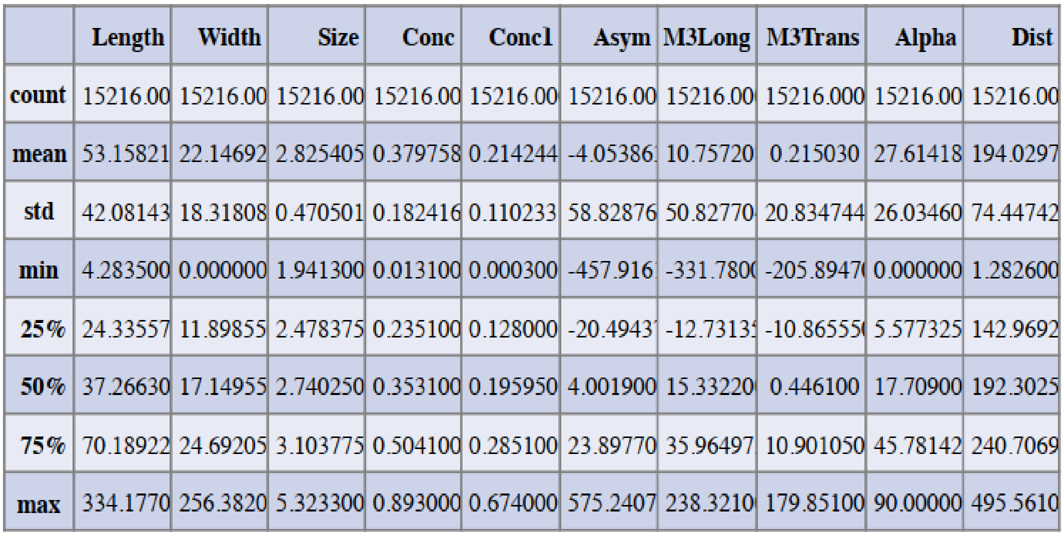
1.1 About the dataset

Nowadays, g-ray astronomy is one of the more solid pillars of astroparticle physics and is turning into an essential tool to study fundamental phenomena in astrophysics, cosmology and high-energy physics that appear in the nonthermal relativistic universe. Cherenkov gamma telescope or MAGIC (Major Atmospheric Gamma Imaging Cherenkov) observes high energy gamma rays, taking advantage of the radiation emitted by charged particles produced inside the electromagnetic showers initiated by the gammas, and developing in the atmosphere. The detector records and allows for the reconstruction of the shower parameter using the imaging technique The reconstruction of the parameter values was achieved using a Monte Carlo simulation algorithm called CORSIKA described in [2]

The dataset has 19020 instances with no missing values. It contains 10 features, which are continuous, and a binary class that indicates an instance to be gamma (signal) or hadron (background). Generally, we are in front of a surpervised machine learning project to detect gamma rays in earth's atmosphere. [1].

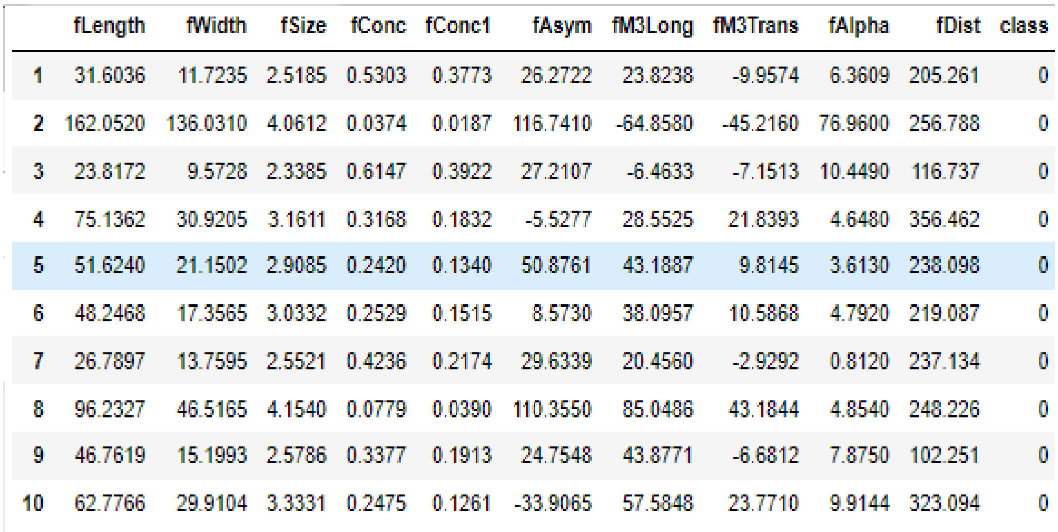
1. fLength: major axis of ellipse [mm]
2. fWidth: minor axis of ellipse [mm]
3. fSize: 10-log of sum of content of all pixels [in phot]
4. fConc: ratio of sum of two highest pixels over fSize [ratio]
5. fConc1: ratio of highest pixel over fSize [ratio]
6. fAsym: distance from highest pixel to center, projected onto major axis [mm]
7. fM3Long: 3rd root of third moment along major axis [mm]
8. fM3Trans: 3rd root of third moment along minor axis [mm]
9. fAlpha: angle of major axis with vector to origin [deg]
10. fDist: distance from origin to center of ellipse [mm]
11. class: g,h gamma (signal), hadron (background)

In figure 1, we provide summary statistics of the dataset's attributes and in figure 2 a sample of ten first instances.



	Length	Width	Size	Conc	Conc1	Asym	M3Long	M3Trans	Alpha	Dist
count	15216.00	15216.00	15216.00	15216.00	15216.00	15216.00	15216.00	15216.000	15216.00	15216.00
mean	53.15821	22.14692	2.825405	0.379758	0.214244	-4.05386	10.75720	0.215030	27.61418	194.0297
std	42.08143	18.31808	0.470501	0.182416	0.110233	58.82876	50.82770	20.834744	26.03460	74.44742
min	4.283500	0.000000	1.941300	0.013100	0.000300	-457.916	-331.780	-205.8947	0.000000	1.282600
25%	24.33557	11.89855	2.478375	0.235100	0.128000	-20.4943	-12.7313	-10.86555	5.577325	142.9692
50%	37.26630	17.14955	2.740250	0.353100	0.195950	4.001900	15.33220	0.446100	17.70900	192.3025
75%	70.18922	24.69205	3.103775	0.504100	0.285100	23.89770	35.96497	10.901050	45.78142	240.7069
max	334.1770	256.3820	5.323300	0.893000	0.674000	575.2407	238.3210	179.85100	90.00000	495.5610

Figure 1: Summary statistics of features.



	fLength	fWidth	fSize	fConc	fConc1	fAsym	fM3Long	fM3Trans	fAlpha	fDist	class
1	31.6036	11.7235	2.5185	0.5303	0.3773	26.2722	23.8238	-9.9574	6.3609	205.261	0
2	162.0520	136.0310	4.0612	0.0374	0.0187	116.7410	-64.8580	-45.2160	76.9600	256.788	0
3	23.8172	9.5728	2.3385	0.6147	0.3922	27.2107	-6.4633	-7.1513	10.4490	116.737	0
4	75.1362	30.9205	3.1611	0.3168	0.1832	-5.5277	28.5525	21.8393	4.6480	356.462	0
5	51.6240	21.1502	2.9085	0.2420	0.1340	50.8761	43.1887	9.8145	3.6130	238.098	0
6	48.2468	17.3565	3.0332	0.2529	0.1515	8.5730	38.0957	10.5868	4.7920	219.087	0
7	26.7897	13.7595	2.5521	0.4236	0.2174	29.6339	20.4560	-2.9292	0.8120	237.134	0
8	96.2327	46.5165	4.1540	0.0779	0.0390	110.3550	85.0486	43.1844	4.8540	248.226	0
9	46.7619	15.1993	2.5786	0.3377	0.1913	24.7548	43.8771	-6.6812	7.8750	102.251	0
10	62.7766	29.9104	3.3331	0.2475	0.1261	-33.9065	57.5848	23.7710	9.9144	323.094	0

Figure 2: Sample of features.

2 Classifiers and metrics

2.1 Logistic regression

The logistic regression model arises from the need to model the posterior probabilities of the K classes through linear functions in x, and at the same time ensuring they sum to one and they in $[0, 1]$. Considering a case where the dependent variables, also called the responses or the outcomes, y_i are discrete and only take values from k = 0,...,$\boldsymbol{K}$-1 (i.e $\boldsymbol{K}$ classes). The goal is to predict the output classes from the design matrix made of $\boldsymbol{n}$ samples, each of which carries $\boldsymbol{p}$ features or predictors. The primary goal is to identify the classes to which new unseen samples belong. In logistic regression, the probability that a data point x_i belongs to a category y_i = 0,1 is given by the so-called logit function (or Sigmoid) which is meant to represent the likelihood for a given event,

$$\mathrm{p}(t) = \frac{1}{1+\exp -t} = \frac{\exp t}{1+\exp t} \tag{1}$$

The model has the form [7]:

$$\begin{aligned}
\log \frac{\Pr(G=1|X=x)}{\Pr(G=K|X=x)} &= \beta_{10} + \beta_1^T x \\
\log \frac{\Pr(G=2|X=x)}{\Pr(G=K|X=x)} &= \beta_{20} + \beta_2^T x \\
&\vdots \\
\log \frac{\Pr(G=K-1|X=x)}{\Pr(G=K|X=x)} &= \beta_{(K-1)0} + \beta_{K-1}^T x
\end{aligned} \tag{2}$$

The model is specified in terms of $K-1$ log-odds or logit transformations (reflecting the restriction that the probabilities sum to one). Even when the model uses the last class as the denominator in the odds-ratios, the choice of denominator is arbitrary in that the estimates are equivalent under thos choice. A simple calculation shows that [7]:

$$\begin{aligned} \Pr(G=k|X=x) &= \frac{\exp\left(\beta_{k0}+\beta_k^T x\right)}{1+\sum_{l=1}^{K-1}\exp\left(\beta_{l0}+\beta_l^T x\right)}, k=1,...,K-1 \\ \Pr(G=K|X=x) &= \frac{1}{1+\sum_{l=1}^{K-1}\exp\left(\beta_{l0}+\beta_l^T x\right)} \end{aligned} \tag{3}$$

and clearly, they sum one.
In order to remark the entire parameter set $\theta = \{\beta_{10}, \beta_1^T, ..., \beta_{(K-1)0}, \beta_{K-1}^T\}$, the probabilities are denoted as $\Pr(G = k|X = x) = p_k(x;\theta)$. When $K = 2$, the model is particularly simple since there is only a single linear function. It is commonly used in bioestatistical applications where binary responses (two classes) happen frequently. For instance, patients survive or die, have a disease or not, or a condition is present or absent [7].

2.2 Decission trees

Of all the learning methods, decision trees are the closest to meet the requirements for serving as an off-the-shelf procedure for data mining. They are relatively fast to construct and they produce interpretable models (if the trees are small). They naturally incorporate mixtures of numeric and categorical predictor variables and missing values. They are invariant under (strictly monotone) transformations of the individual predictors. As a result, scaling and/or more general transformations are not an issue. They are immune to the effects of predictor outliers. They perform internal feature selection as an integral part of the procedure. Hence, they are resistant to the inclusion of many irrelevant predictor variables [7]. These properties of decision trees are mainly the reason that they have become the most popular learning method for data mining. However, trees inaccuracy prevents them from being the ideal tool for predictive learning. They rarely provide predictive accuracy comparable to the best that can be achieved with the data at hand. Boosting decision trees improves their accuracy, often dramatically. At the same time they maintain most of their desirable properties for data mining. Some advantages of trees that are sacrificed by boosting are speed, interpretability, and robustness against overlapping class distributions and especially mislabeling of the training data. A gradient boosted model (GBM) is a generalization of tree boosting that attempts to mitigate these problems, so as to produce an accurate and effective off-the-shelf procedure for data mining [7].

2.3 Random Forests

Random Forests work by training many decision trees on random subsets of the features, then averaging out their predictions [8]. Random forests provide an improvement over bagged trees by way of a random forest small tweak that decorrelates the trees. A number of decision trees is built on bootstrapped training samples. But when building these decision trees, each time a split in a tree is considered, a random sample of m predictors is chosen as split candidates from the full set of p predictors. The split is allowed to use only one of those m predictors. A fresh sample of m predictors is taken at each split, and typically $m \approx \sqrt{p}$, that is the number of predictors considered at each split is approximately equal to the square root of the total number of predictors. In other words, in building a random forest, at each split in the tree, the algorithm is not even allowed to consider a majority of the available predictors [9].

2.4 Support Vector Machine

In order to define support vector machines (SVM), first it's necessary to define the support vector classifiers. This finds linear boundaries in the input feature space. As with other linear methods, it is possible to make the procedure more flexible by enlarging the feature space using basis expansion ssuch as polynomials or splines. Generally linear boundaries in the enlarged space achieve better training-class separation, and translate to nonlinear boundaries in the original space. Once the basis functions $h_m(x), m = 1, ..., M$ are selected, the procedure is the same as before. The support vector classifier is fitted using input features $h(x_i) = (h_1(x_i), h_2(x_i), ..., h_M(x_i)), i = 1, ..., N$, and produces the nonlinear function $\hat{f} = h(x)^T\hat{\beta} + \hat{\beta}_0$. The classifier is $\hat{G}(x) = sign(\hat{f}(x))$ [7].
The support vector machine classifier is an extension of this idea, where the dimension of the enlarged space is allowed to get very large, infinite in some cases. It might seem that the computations would become prohibitive. It would also seem that with sufficient basis functions, the data would be separable, and overfitting would occur.

2.5 Linear and Quadratic Discriminant Analysis

Linear Discriminant Analysis, or LDA for short, works by calculating summary statistics for the input features by class label, such as the mean and standard deviation. These statistics represent the model learned from the training data. In practice, linear algebra operations are used to calculate the required quantities efficiently via matrix decomposition. Predictions are made by estimating the probability that a new example belongs to each class label based on the values of each input feature. The class that results in the largest probability is then assigned to the example. As such, LDA may be considered a simple application of Bayes Theorem for classification.
Quadratic discriminant analysis is quite similar to Linear discriminant analysis except we relaxed the assumption that the mean and covariance of all the classes were equal. Therefore, we required to calculate it separately.

2.6 Confusion Matrix

A confusion matrix is a table that allows the visualization of the performance of a machine learning algorithm. Normally, the rows of the matrix represent the real values obtained for each classes and the columns represent the predicted values of the classes. So, it helps to visualize if the model is confusing or mixing between the two classes.

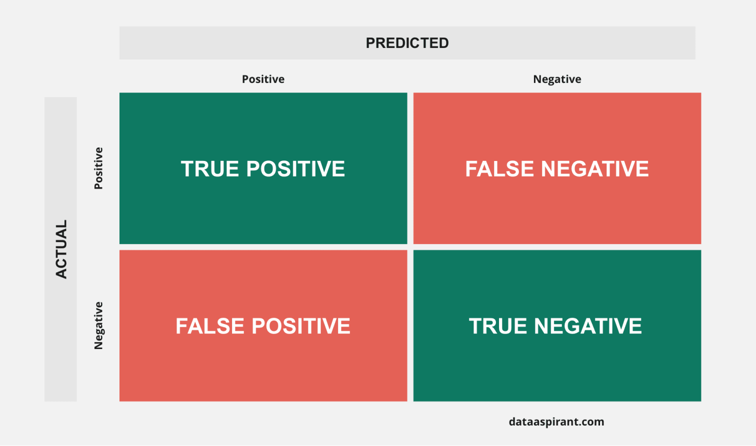


Figure 3: Confusion Matrix.

2.7 The precision-recall curve

The precision-recall curve (PRC) curve is another common tool used with binary classifiers. The precision-recall curve (Fig.4) is used for evaluating the performance of binary classification algorithms. It is often used in situations where classes are heavily imbalanced. Also like ROC curves, precision-recall curves provide a graphical representation of a classifiers performance across many thresholds, rather than a single value such as accuracy and f-1 score. Hence the PR curve plots sensitivity (recall) versus precision. A high area under the curve represents both high recall and high precision, where high precision relates to a low false positive rate, and high recall relates to a low false negative rate. [8].

Average precision (AP) summarizes such a plot as the weighted mean of precisions achieved at each threshold, with the increase in recall from the previous threshold used as the weight:

$$AP = \sum_n ((R_n - R_{n-1})P_n),$$

where P_n and Rn are the precision and recall at the nth threshold. A pair (R_k,P_k) is referred to as an operating point.

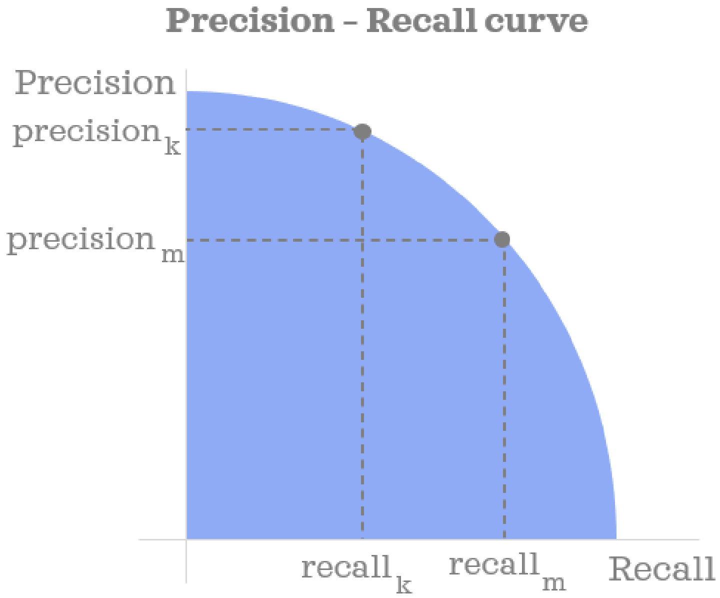


Figure 4: An example of a PRC curve.

3 Methodology

3.1 Data visualization: looking for a strategy

Firtsly, a subplot grid for plotting pairwise relationships in the dataset using pairgrid made and pairwise relationship mapped to those grids. Also, a bar plot of the particles event can be obtain to detect imbalanced in the binary output. The output of this dataset being binary labels it a classification problem , thus validating the use of the proposed methods such as logistic regression and random forest to analyze it.
Secondly, it is important to perform and examine a correlation matrix of all the features. This is done using the heatmap functionality of the seaborn library. The correlation coefficient ranges from -1 to 1. When it is close to 1, it means that there is a strong positive correlation and when it is close to -1, it means that there is a strong negative correlation. Coefficients close to zero mean that there is no linear correlation. In addition, using function ExtraTreeClassifier from sklearn a graph was obtained showing the importance of the features.

3.2 Splitting and scaling the dataset

Regarding the pre-processing part, the dataset was split using **train_test_split** from Scikit-Learn. This procedure involves taking a dataset and dividing it into two subsets. The first subset is used to fit the model (training set) while, the second subset is not used to train the model. Instead, the input element of the dataset is provided to the model, then predictions are made and compared to the expected values (test set). Test set defined as the 25% of the whole data. Thereafter, the training set was again split into a new training and validation set. Validation set capture 15% of the whole data.

Validation sets are taken out of the training set, and used during training to validate the model's accuracy approximately. Since, test set is fully disconnected until the model is finished training, validation set is used to validate model during training.
Therefore, the StandardScaler function in Scikit-Learn was employed to ensure that fo each feature/predictor we study, the mean value is zero and the variance is one(every column in the design/feature matrix). This scaling however has the drawback of not ensuring that we have a particular maximum or minimum in our data set.

3.3 Comparing different classifiers and hyperparameter optimization

The following classifiers are used to train the model and are compared.

- Random Forest.
- Decision Tree
- Logistic Regression
- Support Vector Machine
- Linear Discriminant Analysis
- Quadratic Discriminant Analysis

The approach used serves to objectively search different values for model hyperparameters and choose a subset that results in a model that achieves the best performance on a given dataset. This is called hyperparameter optimization or hyperparameter tuning and is available in the scikit-learn library in python. The result of a hyperparameter optimization is a single set of well-performing hyperparameters. The function GridSearchCv was used fitting on train set and evaluate the performance on a predefined validation set. Class weight was selected Overcoming imbalanced problem.

Table 1: Parameter Grid for each estimator

RF	max features: np.arange(3, 8)	n estimators: [5, 10, 50]	criterion: [gini, entropy]	class weight:[balanced, balanced subsample]
DecTree	max features: np.arange(2, 10)	max depth: np.arange(3, 15)	criterion:[gini, entropy]	class weight:[balanced ,None]
LogReg	C :np.logspace(-3,3,20)	penalty :[l2, l1,elasticnet, none]	solver :[newton-cg, lbfgs, liblinear, sag, saga]	class weight: [balanced,None]
SVM	C: [0.1, 1, 10, 100, 1000	gamma: [10, 1, 0.1, 0.01, 0.001, 0.0001]	kernel: [rbf]	class weight:[balanced, None]
LDA	shrinkage: [auto, float, None]	tol:[0.0001,0.001,0.01,0.1]	solver: [svd, lsqr, eigen]	
QDA	reg param:[0.01,0.0,0.1,0.2,1.0,10.0]	tol:[0.0001,0.001,0.01,0.1		

3.4 Evaluating the performance for best classifiers

The simple classification accuracy is not meaningful for this data, since classifying a background event as signal is worse than classifying a signal event as background. In addition, accuracy lacks on imbalanced data. In order to deal with imbalance between classes other metrics was used as F-score and precision. The best classifiers were selected and compared with each other as predefined in the previous step. For comparison of different classifiers a precision recall curve was used. In hyperparameters tuning average precision was used as scoring parameter.

3.5 Further Analysis: Feature Selection

Feature selection is the process of reducing the number of input variables when developing a predictive model. It is desirable to reduce the number of input variables to both reduce the computational cost of modeling and, in some cases, to improve the performance of the model. Many models, especially those based on regression slopes and intercepts, will estimate parameters for every term in the model. Because of this, the presence of non-informative variables can add uncertainty to the predictions and reduce the overall effectiveness of the model.

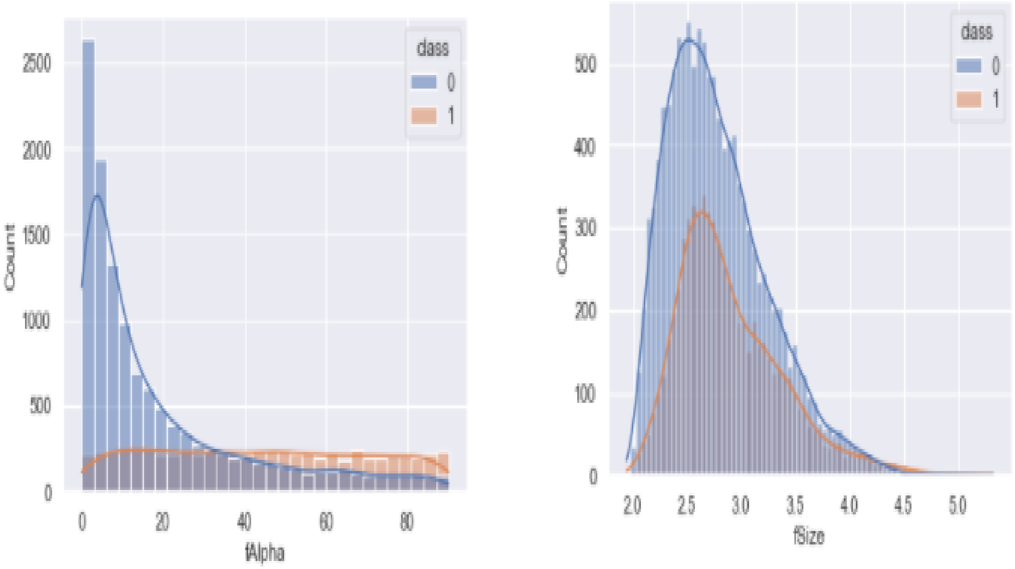


Figure 5: fAlpha and fSize distribution

4 Results

4.1 Data visualization: looking for a strategy

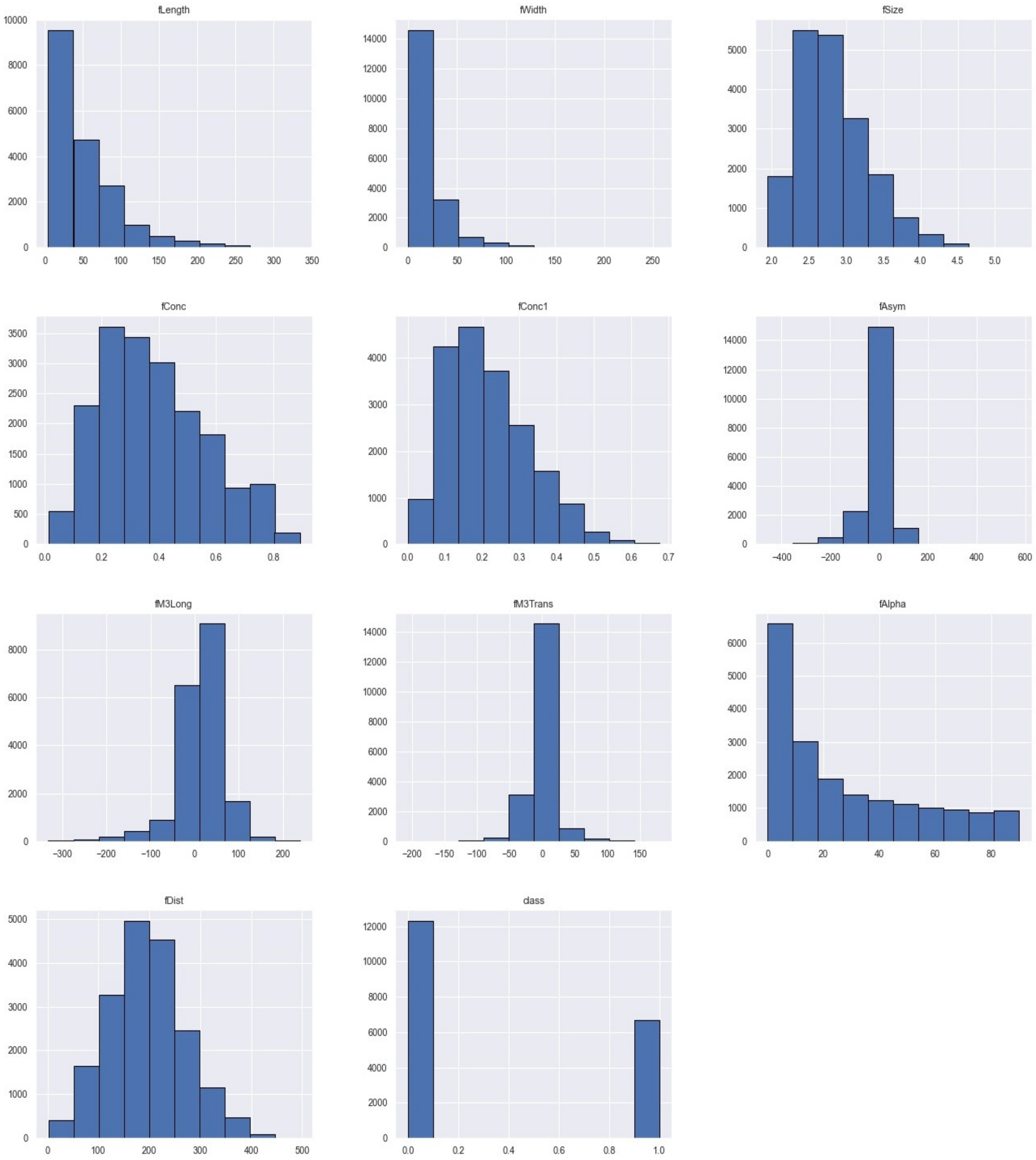


Figure 6: Features of the data set.

As illustrated in Figure 6, most of the features are positive skewed, except from 'fAsym' and 'fM3trans' with normal distribution and 'fM3Long' that is negative skewed. That means that there are possible cut offs. Classifiers such as, Random Forest and Decision tree perform well in this situation.

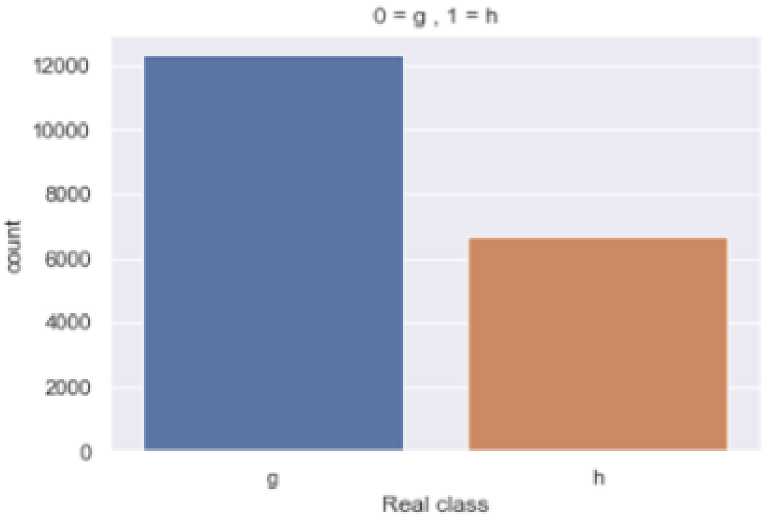
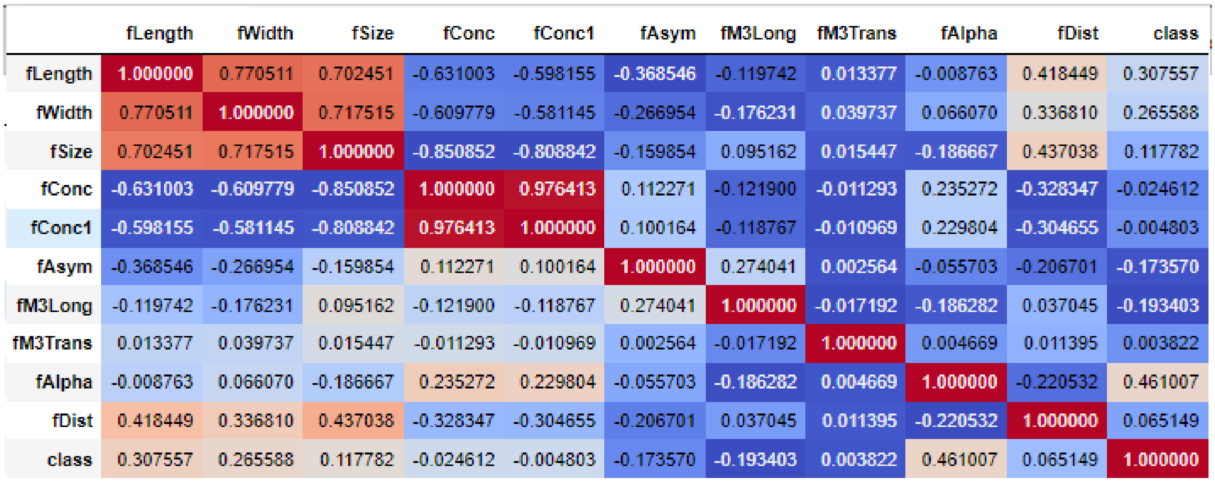


Figure 7: Distribution of the particles.

As it is shown in the Fig. 7, the dataset is unbalanced with 12332 and 6688 instances for gamma and hadron, respectively. For technical reasons, the number of h events is underestimated. In the real data, the h class represents the majority of the events. Class imbalanced should be taken under consideration in hyperparameter tuning.
As it can be seen in the correlated matrix (Fig.8) and the (Fig.9) the target has a strong correlation with the following features: fAlpha, fLength, fSize and fWidth.

- The most important feature of this dataset is the angle of major axis with vector to origin (fAlpha). In the majority of instances for angles close to zero the particle is gamma ray. On the contrary, close to vertical angles the event is label as background.
- Mentioning that fLength and fWidth has a significant impact on the output. This features represents the major and minor axis of the ellipse of Cherenkov light. That means that are highly correlated. So the useful information can be obtained by one of them.
- In physics problems like that is crucial to estimate the Energy. The simplest energy estimator of our feature is the number of photons (fSize).
- Finally, fDist and fM3Long are useful tools in predictions.



	fLength	fWidth	fSize	fConc	fConc1	fAsym	fM3Long	fM3Trans	fAlpha	fDist	class
fLength	1.000000	0.770511	0.702451	-0.631003	-0.598155	-0.368546	-0.119742	0.013377	-0.008763	0.418449	0.307557
fWidth	0.770511	1.000000	0.717515	-0.609779	-0.581145	-0.266954	-0.176231	0.039737	0.066070	0.336810	0.265588
fSize	0.702451	0.717515	1.000000	-0.850852	-0.808842	-0.159854	0.095162	0.015447	-0.186667	0.437038	0.117782
fConc	-0.631003	-0.609779	-0.850852	1.000000	0.976413	0.112271	-0.121900	-0.011293	0.235272	-0.328347	-0.024612
fConc1	-0.598155	-0.581145	-0.808842	0.976413	1.000000	0.100164	-0.118767	-0.010969	0.229804	-0.304655	-0.004803
fAsym	-0.368546	-0.266954	-0.159854	0.112271	0.100164	1.000000	0.274041	0.002564	-0.055703	-0.206701	-0.173570
fM3Long	-0.119742	-0.176231	0.095162	-0.121900	-0.118767	0.274041	1.000000	-0.017192	-0.186282	0.037045	-0.193403
fM3Trans	0.013377	0.039737	0.015447	-0.011293	-0.010969	0.002564	-0.017192	1.000000	0.004669	0.011395	0.003822
fAlpha	-0.008763	0.066070	-0.186667	0.235272	0.229804	-0.055703	-0.186282	0.004669	1.000000	-0.220532	0.461007
fDist	0.418449	0.336810	0.437038	-0.328347	-0.304655	-0.206701	0.037045	0.011395	-0.220532	1.000000	0.065149
class	0.307557	0.265588	0.117782	-0.024612	-0.004803	-0.173570	-0.193403	0.003822	0.461007	0.065149	1.000000

Figure 8: Correlation matrix.

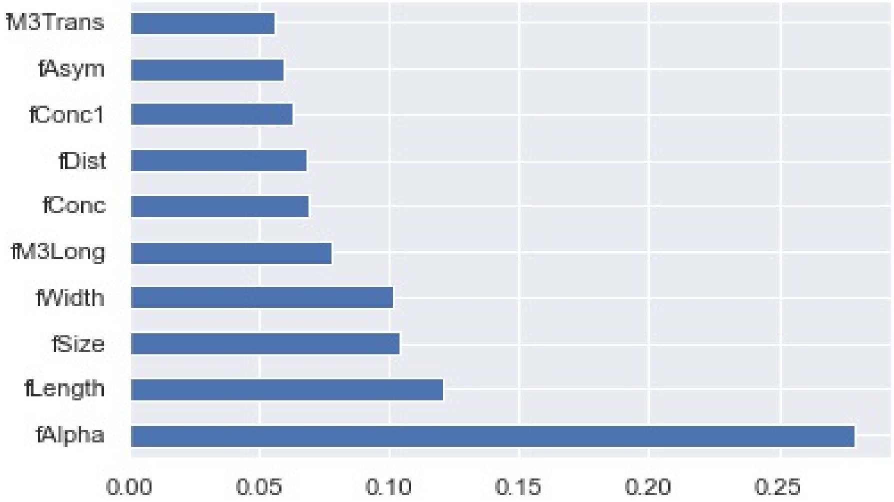


Figure 9: Feature importances

4.2 Comparing different classifiers and hyperparameter optimization

In this step the best performance among the classifiers was determined by hyperparameters tuning. The results was obtained by performing classifiers are listed in the table 2

Table 2: GridSearchCV results

Estimators	Best hyperparameters				Best score
RF	max features: 4	n estimators:50	criterion: entropy	class weight: balanced_subsample	0.88
DecTree	max features: 8	max depth: 5	criterion: entropy	class weight:balanced	0.80
LogReg	C : 0.16	penalty : l1	solver: saga	class weight: None	0.78
SVM	C: 100	gamma: 0.01	kernel: rbf	class weight: balanced	0.87
LDA	shrinkage: auto	tol: 0.0001	solver: lsqr		0.76
QDA	reg param:0.0	tol: 0.0001			0.76

4.3 Evaluating the performance for best classifier

A more overall estimation of the results was carried out using the metrics as shown in Table 3

Table 3: Estimators Perfomance on Validation set

Estimators	F-score	Precision	Average Precision	Accuracy
RF	80.00%	87.73%	**73.61%**	87.35%
DecTree	75.80%	76.37%	66.23%	82.98%
LogReg	66.14%	75.60%	59.04%	78.68%
SVM	80.28%	81.69%	**71.94%**	86.27%
LDA	64.15%	75.88%	57.90%	78.00%
QDA	61.62%	82.83%	58.68%	78.35%

A visual comparison between the best classifiers made by using precision recall curves with a threshold at 0.4. As it shown in figures (10 , 11, 12), Random Forest and SVC classifiers performing very well.

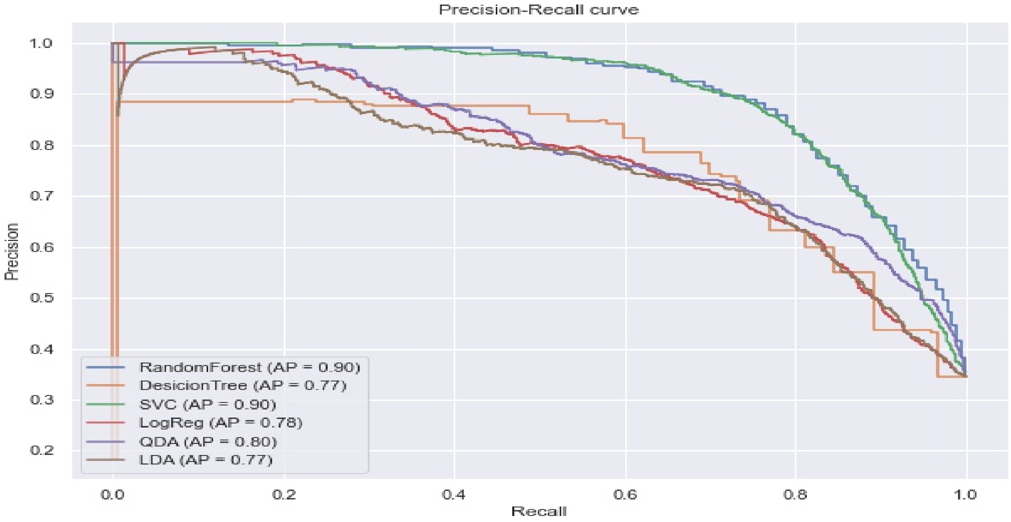


Figure 10: Precision-Recall curve.

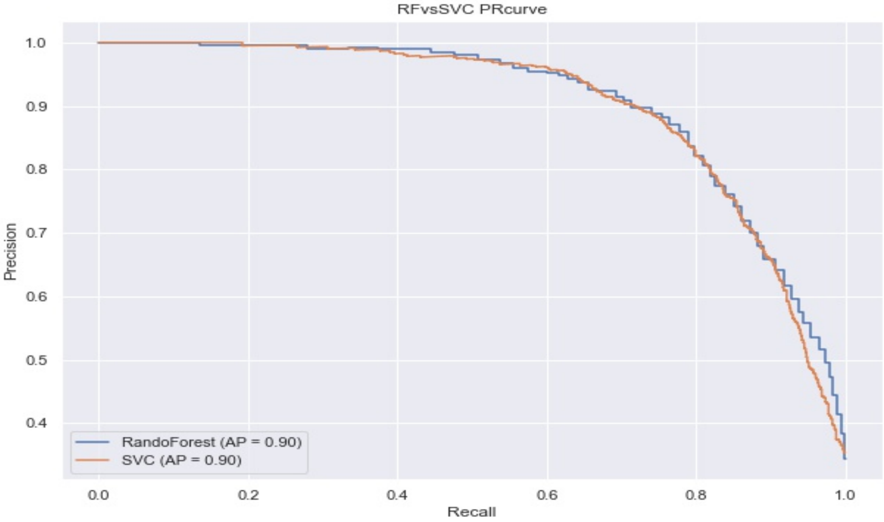
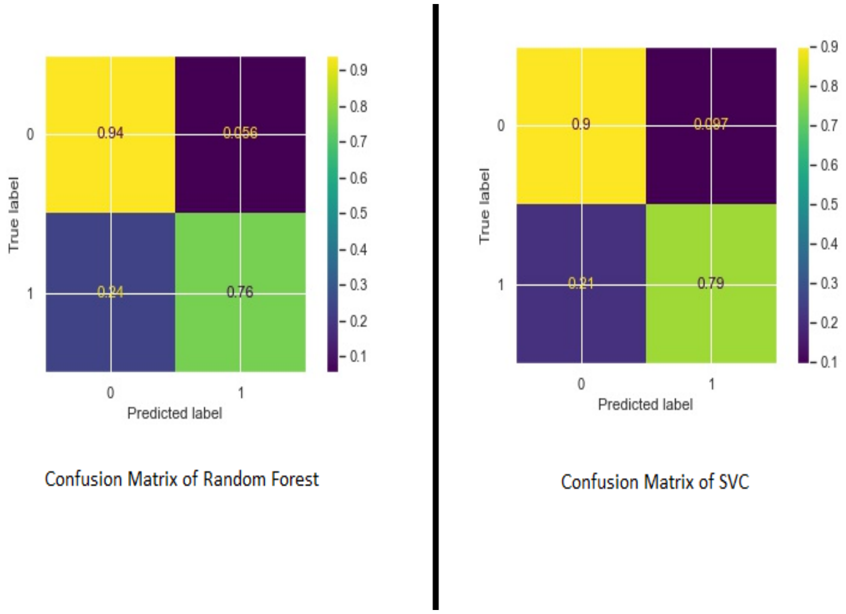


Figure 11: Random Forest vs SVC.



Confusion Matrix of Random Forest

Confusion Matrix of SVC

Figure 12: Random Forest vs SVC.

4.3.1 Final Model

As a conclusion, the best model for gamma detection in MAGIC gamma telescope is random forest for the hyperparameters shown in table 4. Also, this model fitting on train set and testing in test set. As a result, the Precision Recall curve is shown in figure 13.

Table 4: Final model

Estimator	max features	n estimators	criterion	class weight
RandomForest	3	50	entropy	balanced

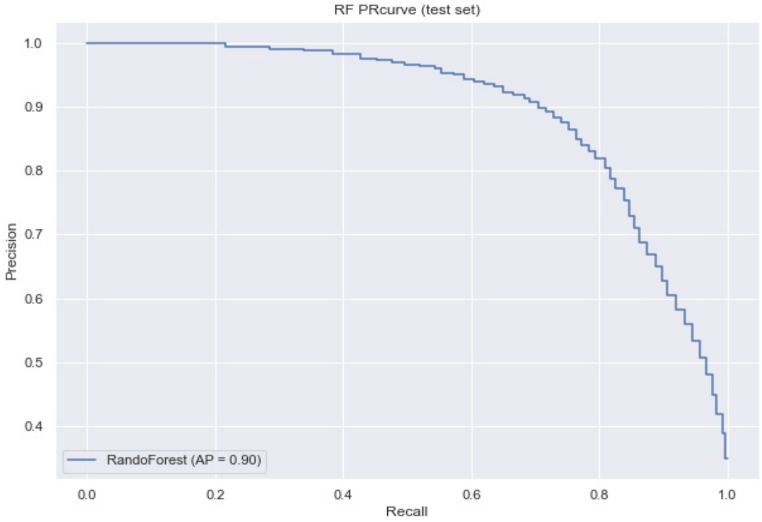


Figure 13: Random Forest Curve.

4.4 Further Analysis: Feature selection

Most of classifiers make feature selection by reducing the dimensions, such as Random Forest, Decision Tree and Logistic Regression. In the current analysis, feature selection is done manually, advising results from correlation matrix (Fig 8) and feature importance (Fig 9). The features that be selected are:

- fAlpha
- fLength
- fSize
- fM3Long
- fDist

The results of classifier's evaluation after the feature selection is shown in the precision curve below (Fig 14)

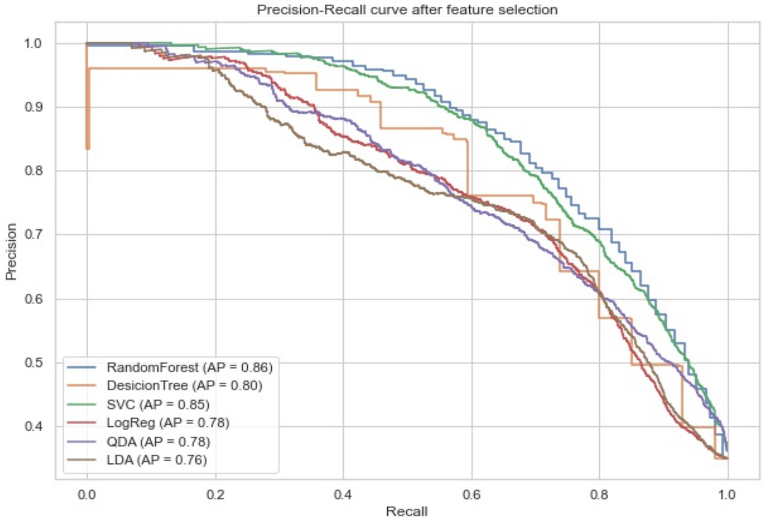


Figure 14: Perfomance after feature selection

5 Conclusions

- Given dataset classes are not disributed randomly. So shuflle the data before splitting is necessary.
- Scaling is important to overcome skeweness and outliers. In addition for classifiers such as logistic regression, scaling is required.
- Precision recall curve provides the most trusted results considering the imbalanced classes and the sensitivity at false positives.
- All classifiers provides excellent results after hyperparameter optimization. However Random Forest and SVC are the better estimators
- Results after feature selection are not so different. Therefore, QDA shows a slight improvement after feature selection
- Useful conclusion can be obtain by feature analysis. For Instance, if the angle of major axis with vector to origin (fAlpha) is close to zero, the particle possibly is gamma ray.

References

[1] UCI Machine Learning Repository: MAGIC Gamma Telescope Data Set [Internet]. Archive.ics.uci.edu. 2021 Available from: MAGIC site:archive.ics.uci.edu/ml site:ics.uci.edu

[2] Bock, R.K., Chilingarian, A., Gaug, M., Hakl, F., Hengstebeck, T., Jirina, M., Klaschka, J., Kotrc, E., Savicky, P., Towers, S., Vaicilius, A., Wittek W. (2004). Methods for multidimensional event classification: a case study using images from a Cherenkov gamma-ray telescope. Nucl.Instr.Meth. A, 516, pp. 511-528.

[3] P. Savicky, E. Kotrc. Experimental Study of Leaf Confidences for Random Forest. Proceedings of COMPSTAT 2004, In: Computational Statistics. (Ed.: Antoch J.) - Heidelberg, Physica Verlag 2004, pp. 1767-1774.

[4] J. Dvorak, P. Savicky. Softening Splits in Decision Trees Using Simulated Annealing. Proceedings of ICANNGA 2007, Warsaw, (Ed.: Beliczynski et. al), Part I, LNCS 4431, pp. 721-729.

[5] Lecture notes, 1999-2021, Morten Hjorth-Jensen. Released under CC Attribution-NonCommercial 4.0 license

[6] Bishop, C. *Pattern Recognition and Machine Learning.* Springer: Singapore, 2006.

[7] Hastie, T., et al. *The Elements of Statistical Learning. Data mining, interference and prediction.* Springer, 2nd edition, 2016.

[8] Geron, A. *Hands-On Machine Learing With Scikit-Learn and Tensor Flow.* O'Reilly Media, 2017.

[9] Hastie, T., et. all. *An Introduction to Statistical Learning.* Springer Science+Business Media, New York, (2017).