Tesina
MAGIC Gamma Telescope Data Set

Armando La Rocca s279401
Mathematics in machine learning
2019/2020

October 1, 2020

1 Introduction

In this tesina the MAGIC Gamma dataset is analyzed and the classification problem related to the recognition of Gamma Ray is performed.
In the fist part is analyzed the background related to the dataset and how data are collected. Then a brief description of the dataset is presented.
In the second part, data are cleaned, the correlation between features is analyzed and the management of the outliers is considered. Then the pre-preocessing step with the validation method, the sampling technique and the normalization of the data are described.
Finally, the classification, that is the core part of the tesina, is performed. Firstly, the metrics used to evaluate the classifiers are briefly explained and then the classification is performed using four algorithms: KNN, SVM, Decision Tree and Random forest. The last part is related to the comparison of the results obtained with the different classifiers.
The code is provided at:
`https://github.com/armando-larocca/Tesina_MML`

2 Data exploration

2.1 MAGIC Gamma Telescope Data Set

The dataset used in this tesina is related to the MAGIC telescope that is able to identify and observe high energy Gamma ray using the imaging technique. Particularly the telescope analyze the Cherenkov radiation that can be produced by Gammy ray entering in the atmosphere or by some other atmospheric phenomenon. The telescope catches the radiation with a complex system of mirrors that reflect the information producing an image called "shower".

The shower is firstly pre-processed by the telescope and the features of the dataset are obtained. The goal of this analysis is understand which shower was produced by gamma ray (signal) and which one by other background atmospheric phenomena (background).
Data of this specific dataset are generated by a

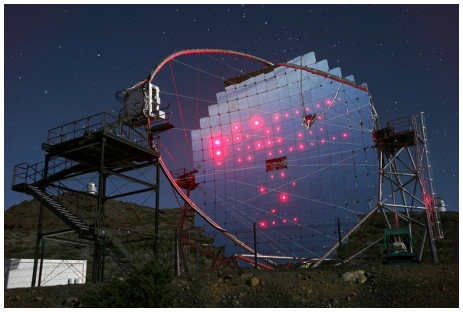


Figure 1: The MAGIC telescope

Monte Carlo program named Corsika and are related to event with no more than 50 GeV.

2.2 Data visualization

The dataset is composed by 10 quantitative features and the categorical labels that allow to discriminate between gamma ray (g) and background (h).
Below is presented a brief description of the features:

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

In the dataset there are not missing values and in the following table is represented the description of the data distribution.

	1.fLength	2.fWidth	3.fSize	4.fConc	5.fConc	6.fAsym	7.fM3Long	8.fM3Trans	9.fAlpha	10.fDist
count	19020.000000	19020.000000	19020.000000	19020.000000	19020.000000	19020.000000	19020.000000	19020.000000	19020.000000	19020.000000
mean	53.250154	22.180966	2.825017	0.380327	0.214657	-4.331745	10.545545	0.249726	27.645707	193.818026
std	42.364855	18.346056	0.472599	0.182813	0.110511	59.206062	51.000118	20.827439	26.103621	74.731787
min	4.283500	0.000000	1.941300	0.013100	0.000300	-457.916100	-331.780000	-205.894700	0.000000	1.282600
25%	24.336000	11.863800	2.477100	0.235800	0.128475	-20.586550	-12.842775	-10.849375	5.547925	142.492250
50%	37.147700	17.139900	2.739600	0.354150	0.196500	4.013050	15.314100	0.666200	17.679500	191.851450
75%	70.122175	24.739475	3.101600	0.503700	0.285225	24.063700	35.837800	10.946425	45.883550	240.563825
max	334.177000	256.382000	5.323300	0.893000	0.675200	575.240700	238.321000	179.851000	90.000000	495.561000

Figure 2: Dataset Description

An important aspect that is considered for this analysis is that the dataset is unbalanced. In fact, as represented in Figure 3. the number of instances of gamma is higher than background even if the number of background events is more frequently observed in real life.

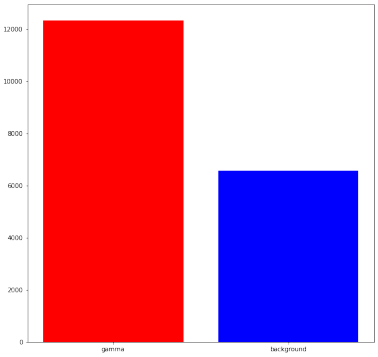


Figure 3: Sample per class

Then is represented a pair plot where is clear the unbalancing between the two classes. Another interesting aspect is that the two classes data have a quite similar distribution.

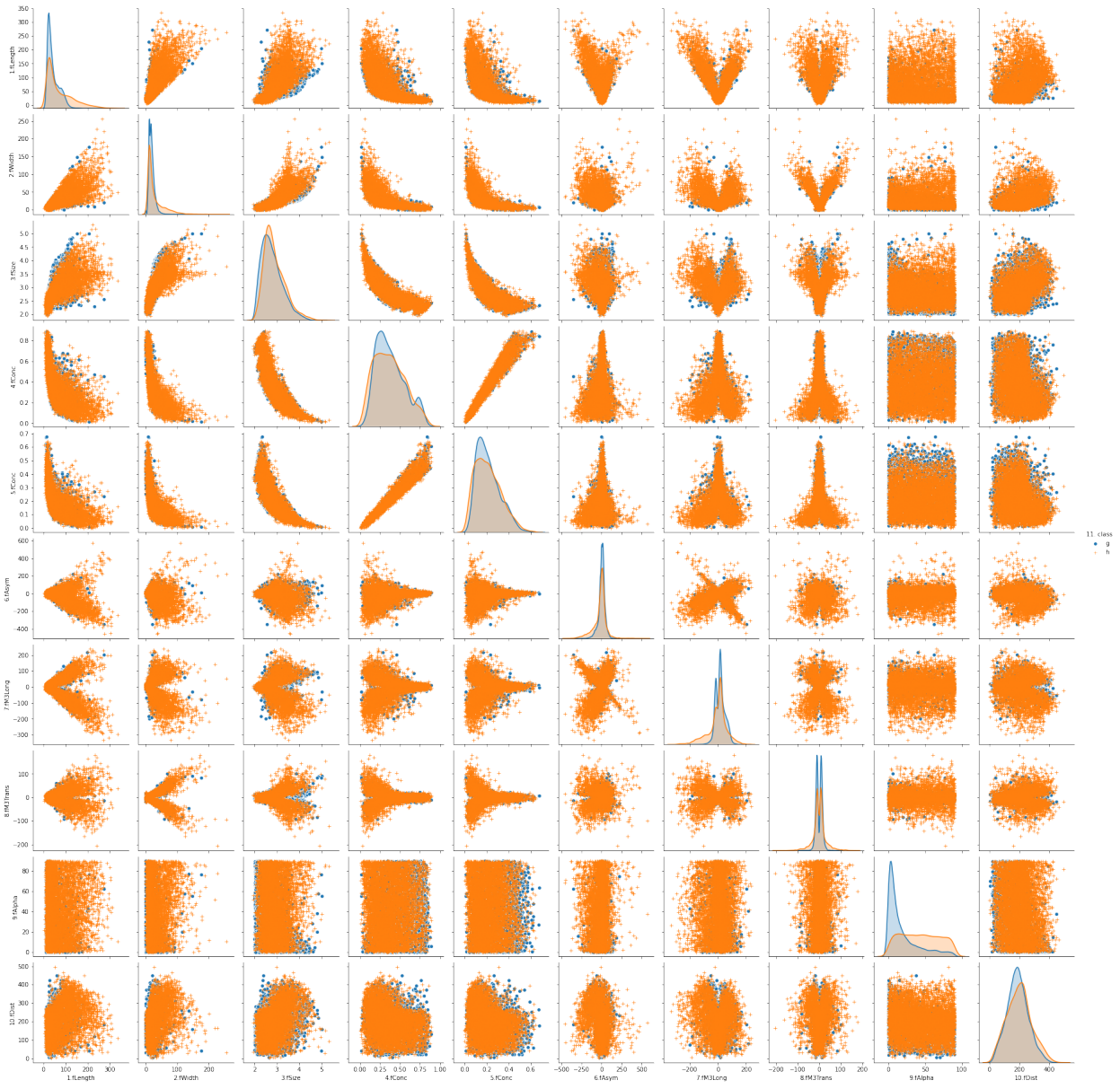


Figure 4: Sample per class

3 Data cleaning

3.1 Duplicates

In the analysis are founded 115 duplicates that correspond to the 0.6% of the entire data. The duplicates are removed as first step of the cleaning process.

3.2 Correlation

Then was performed an analysis of the correlation between the features. Particularly in this case is calculated the Spearman correlation coefficient that is a measure of the monotonic correlation between two features and is based on the ranks. The correlation coefficient is calculated considering the following formula and can assume values between $[-1, 1]$.

$$\rho = 1 - \frac{6 \sum_i d_i^2}{n(n-1)}$$

where $d_i = x_i - y_i$ is the difference between the ranks of the two features of the instance i and n is the number of samples.

In Figure 5 can be seen that the features "4.Fconc" and "5.Fconc" have an high value of correlation that is around 0.99.For this reason and also considering the similarity of this features, is decided to drop out the feature "5.fConc".

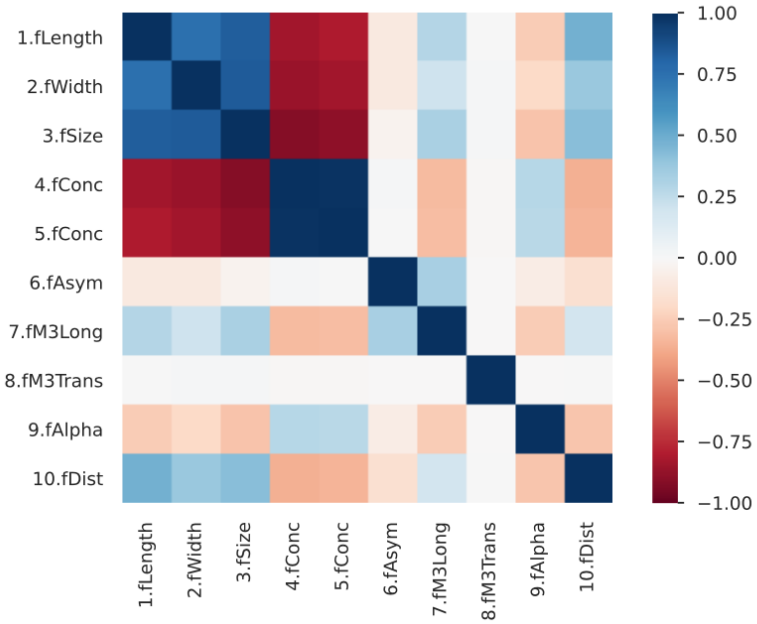


Figure 5: Correlation matrix

3.3 Outliers management

In this analysis is not included the management of the outliers because the shower image is pre-processed by the telescope itself in order to excluded images that are not useful or correct for the following study. This phase is prior to the creation of the dataset because the excluded showers can be caused by bad weather conditions or very particular atmosphere perturbations. For this reason all the elements in the dataset are considered relevant. [1]

4 Data pre-processing

First of all, the dataset is divided in training set and test set. The training part accounts the 80% of the total data while the test split the last 20%.

4.1 K-fold Cross Validation

The evaluation of a machine learning algorithm can be very difficult because the metric that is used to evaluate the algorithm can be affected by biases related to the data that are in the validation set. Particularly the validation is a common operation made to tune the hyperparameters and improve the results.
For this reason the K-fold validation is performed. The main strategy of this approach is to divide the training set in k splits. The training and the validation operations are performed k time using a different split as validation and the others as training. Each time is obtained a score and the final result of the cross validation is the mean of the k scores obtained.

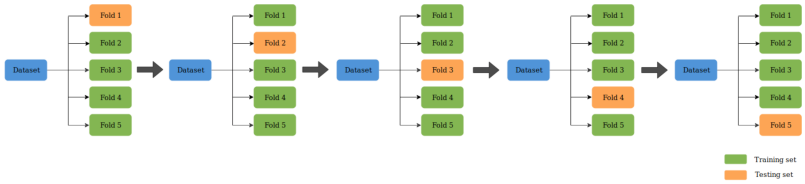


Figure 6: K-fold validation scheme

In this analysis is performed a 5-fold validation as reported in Figure 6

4.2 Sampling methods

As said before, the dataset is unbalanced because the number of gamma instances is higher than the background ones. This can affect the performances of the classifier increasing the misclassification of the less represented class. To avoid this behaviour is performed a sampling procedure to balance the instances of each class in the training set. In this analysis are presented two sampling methods.

Random undersampling
The random undersampling method pick n samples randomly from the class data that is more represented (gamma class in this case), where n is the number of instances of the less represented one (the background class). In contrast, the data related to the less represented class are taken without

changing.
Usually the oversampling is a more popular strategy but in this case the number of training instances is still sufficient also to perform the undersampling.

Oversampling and undersampling mix
Another possible solution to deal with unbalanced dataset is a combination of oversampling and undersampling approach. In this case will be used a combination of Tomek links and SMOTE techniques.

Firstly, is applied the Tomek links technique that is the undersampling part. This is an approach that works finding the Tomek links between the different classes. Considering two samples of different classes x, y they are a Tomek's link if for each sample z:

$$d(x, y) \leq d(x, z) \; and \; d(x, y) \leq d(y, z)$$

Overall, ff two samples of different classes are nearest neighbors there is a Tomek link. The data connected by Tomek links are rejected.

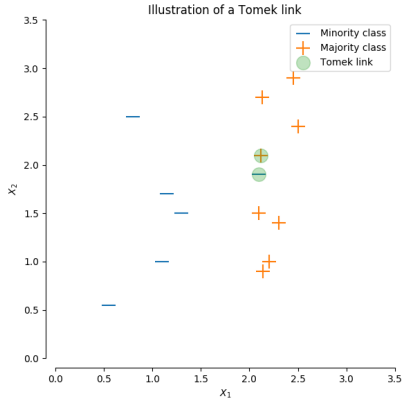


Figure 7: Tomek links example

The SMOTE techniques is an oversampling approach that synthesize new samples of the of the less represented class starting from the existing one. This algorithms use the concept of k-nearest neighbors and can be well explained considering different steps :

- A point x_1 of the less represented class is randomly chosen.
- Another point x_2 is randomly chosen between the K nearest neighbors.
- λ, that is a value between [0,1], is randomly chosen.
- The new synthetic point takes place between the two considered points with a distance of $\lambda \cdot d(x1, x2)$ from the original point $x1$. [$d(x1, x2)$ is the distance between $x1x2$]

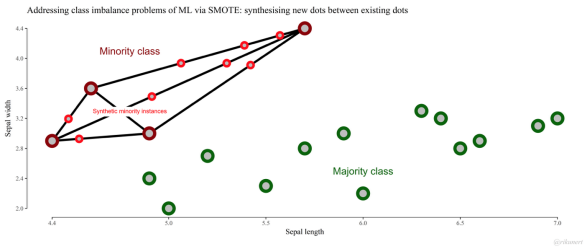


Figure 8: Smote oversampling example

4.3 Normalization

As represented in Figure 2 and 9 can be seen that different features have different scale. This is quite evident, for instance, comparing the feature 4 with the feature 10. These data with different scales can affect the performances of some classifiers and in general the normalization is a common and an important practice in machine learning to improve results.

Min-max Normalization
In this analysis is used the min-max normalization that scales the values in a range between [0,1] in the following way:

$$x' = \frac{x - min(x)}{max(x) - min(x)}$$

where x' is the normalized value while x is the original one.

Both training data and test/validation sets are normalized but on the test data is applied the same transformation that belongs from training distribution. This approach is made because the test and validation data are considered as unknown.

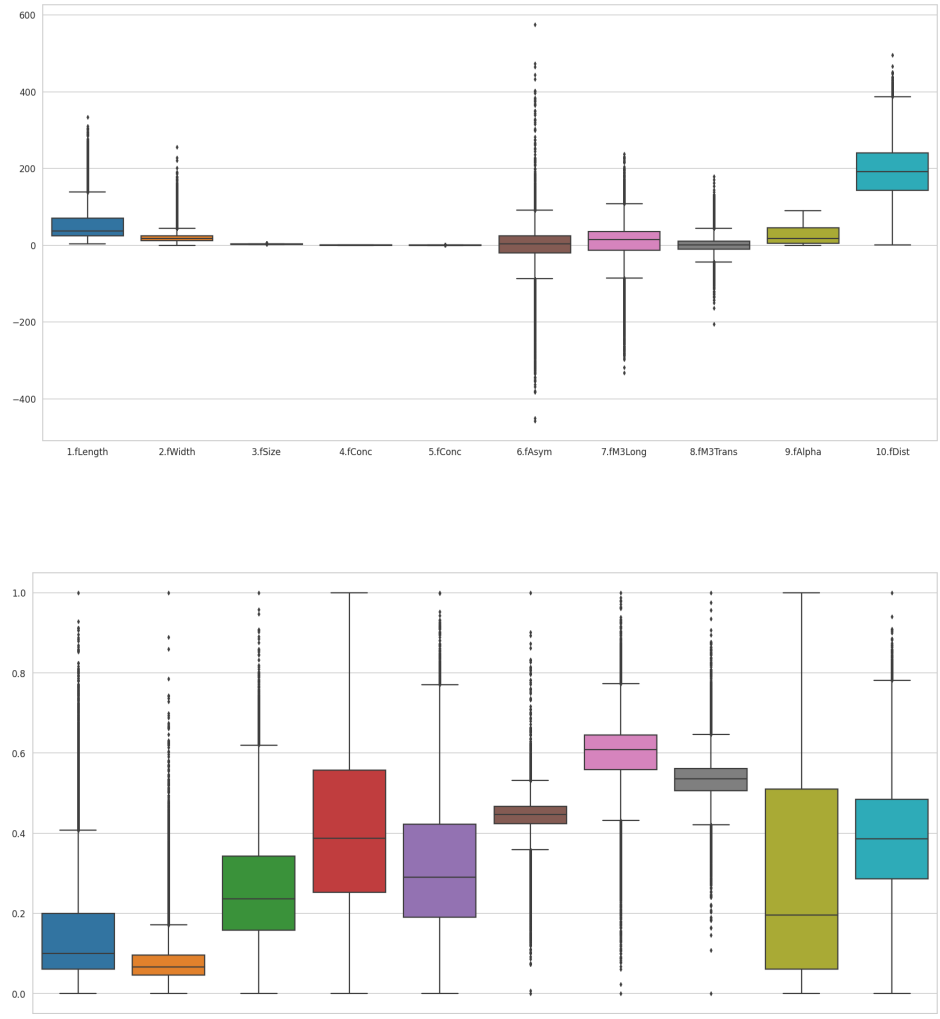


Figure 9: Example of data before and after normalization

5 Classification

In this section different classification algorithm are used on the presented dataset. First of, all the metrics that will be used to evaluate the classifiers are presented. The algorithms are briefly introduced and the 5-fold validation is performed in order to obtain the best hyperparameters. Each classifier is then evaluated on the test and, as last part part, the results are compared in order to identify the best classifier for this problem. This analysis is performed twice using the two sampling methods proposed in the last section. This approach is made also to better understand how different sampling approaches can change the performances of the classification.
The following flowchart represents very briefly the process followed for the analysis.

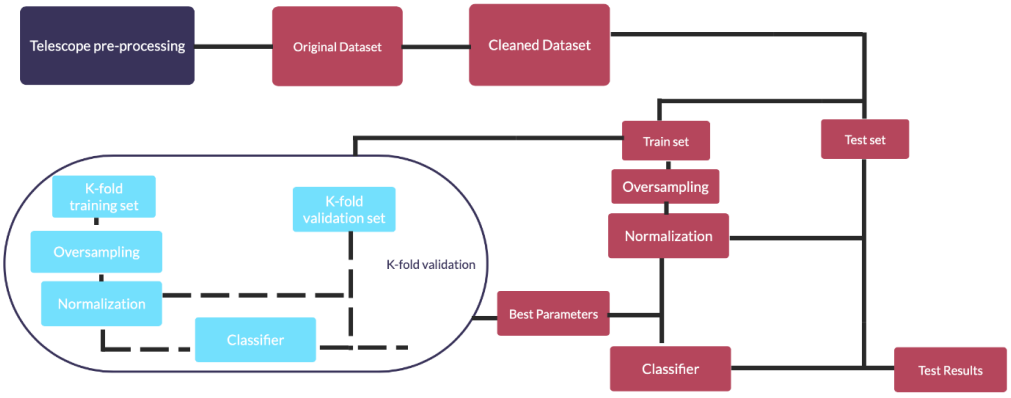


Figure 10: Analysis flowchart

5.1 Evaluation metrics

For the evaluation of the results are considered different metrics in order to have a clear view of the results.

Accuracy
The accuracy is one of the most common and intuitive metric of evaluation and for this reason is employed also in this analysis.

Overall the accuracy is the fraction of predictions our model got right and can be calculated as:

$$Acc = \frac{\#samples\_correctly\_classified}{\#of\_allsamples}$$

Confusion matrix
This graph is very useful to evaluate the performances of a classification algorithm because in contrast with the accuracy present a more clear overview of all the results. Considering, for instance,a binary classification between Positive and Negatives. The Confusion matrix presents four numbers :

- True Positive : The positive instances correctly classified as positive
- False Negative : The positive instances incorrectly classified as negative
- False Positive : The negative instances incorrectly classified as positive
- True Negative : The negative instances correctly classified as negative

		Actual class	
		P	**N**
Predicted class	**P**	**TP**	FP
	N	FN	**TN**

Figure 11: Confusion Matrix

The confusion matrix is a very powerful tool and from it all the metrics can be obtained.

F1 score
When an unbalanced dataset is used, an important metric to evaluate the accuracy on the test is the F1 score. This value is obtained as the harmonic mean of precision and recall. For the description of the following metrics will be used the values explained for confusion matrix.

The precision is a measure of the correctly identified as positive in the all positive predicted.

$$Precision = \frac{TP}{TP + FP}$$

The recall is a measure of the correctly identified as positive in the all the really positive.

$$Recall = \frac{TP}{TP + FN}$$

The F1, in contrast with the accuracy, gives more relevance to the True Negative and False positive and for this reason is more used in case of unbalanced classes.

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

This metric will be used as discriminant parameter choosing the best hyperparameters of the algorithms.

Receiver Operator Characteristic

The ROC curve is a specific plot that shows the performances of a binary classifier with different tresholds. The graph has two parameters that are the True Positive rate and the the False False positive rate :

$$TPR = Recall = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{FP + TN}$$

The ROC curve is a good way to compare classifiers when the accuracy parameters is not enough. Generally, bigger is the area under the curve (AUC) better are the performances.

In this specific analysis this method is used, as suggested by the dataset description, because classify a background event as signal is worse than classify a signal event as background. For this reason are considered only the points with FPR under the threshold of 0.2. [2]

This approach will be used in the last of the analysis to make a comparison between all the classifiers on the test set.

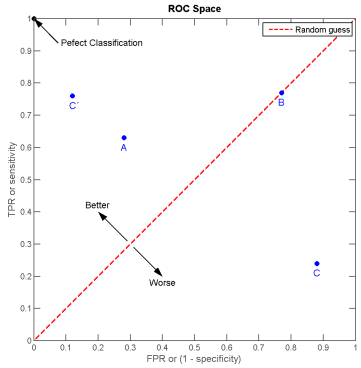


Figure 12: Roc space representation

5.2 K-Nearest Neighbors

The KNN is a supervised learning algorithm that can be used for both classification and regression. Given a training set (X, Y),it classify a new instance x considering the K closest point of the training set to x, assigning the class of the most recurrent one. The distance is a parameter of this classifier and in this case the euclidean distance is chosen.
K is an important hyperparameter in the classification and the right choice of this value depends by the data. Particularly with high values of K the classification is less sensitive to the noise in the data while low values of K is more sensitive.

Validation Results
The validation is made to chose the best values of K. The choice is based on the best value of the F1 score. In the following graphs are plotted the accuracy and the F1 score over different values of K.

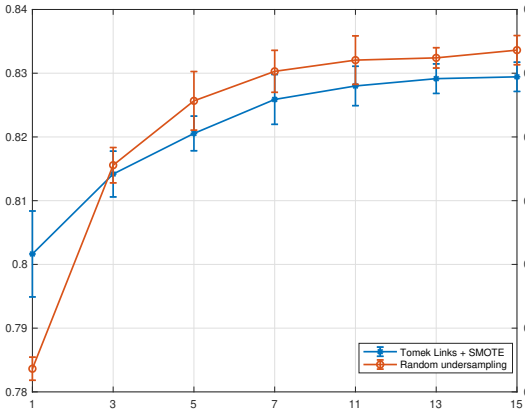


Figure 13: Accuracy

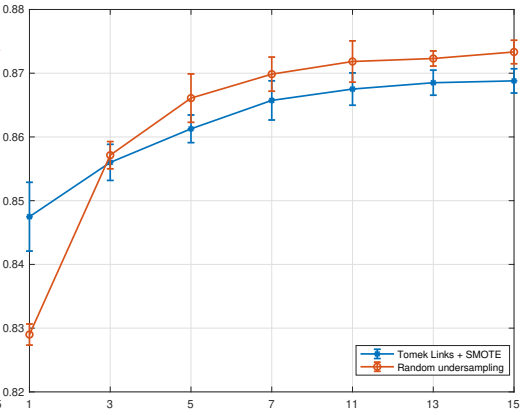


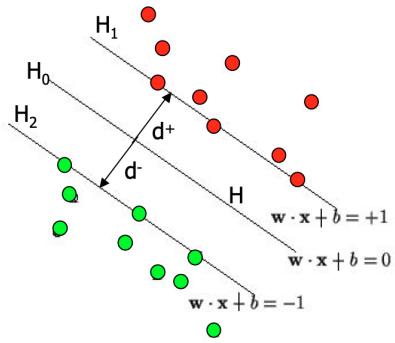
Figure 14: F1 score

It's clear that in this case the undersampling method is more efficient than the mix of over/undersampling. However with both the strategy the trend is the same for both the metrics. In fact K=15 is the best value between the one that have been analyzed and the F1 score reached a value of 86.88%.

5.3 Support Vector Machine

The SVM is a supervised learning algorithm. Overall, the main goal of this algorithm is to find an hyperplane (or a set of hyperplanes in case of multi-classification) that divides the training data belonging from different classes. The classification of new instances is performed considering their position related to the hyperplane.
The choice of the hyperplane can be very different and depends on the data that are on the boarder of the distributions named support vectors.

H_o : hyperplane $w \cdot x + b = 0$.
d_+, d_-: distances from the support vectors to the hyperplane;



The hard margin SVM tries to maximizes the margin to make a more robust classification. However this approach assume linearly separable data, that is a strong assumption. In case of noisy data the soft margin SVM can be used. This approach relaxes the assumption of linearly separable data and it is used for the following analysis.
More formally, the hyperplane is chosen in the following way :

$$\min_{w} \frac{1}{2} w^T \cdot w + C \sum_{i=1}^{n} max(0, 1 - w^T x_i)$$

where the second term is the hinge loss and C is a regulation term that is a trade off between a large margin and a small loss. High values of C make more relevant the loss and in this sense the algorithm is more careful to avoid mistakes. Low values of C gives priority to have larger margin taking less care to misclassification.

This is a quadratic optimization problem can be solved using the gradient descent. Firstly w is initialized with a random value w_0. Considering :

$$J(w) = \frac{1}{2} w^T \cdot w + C \sum_{i=1}^{n} max(0, 1 - w^T x_i)$$

for $t = 0, 1..N$ w is updated :

$$w^{(t+1)} = w^t - \gamma \nabla J(w)$$

where γ is the learning rate

In the case in which data are non linearly separable the kernel trick can be used. The kernel is a function, for a mapping ψ, that implements inner products in the feature space.

$$K(x, x') =< \psi(x), \psi(x') >$$

Using this trick is possible to make training and prediction using K alone without apply at all ψ.

There are different kind of kernel functions that can be used but in this case is used the Gaussian kernel (RBF) :

$$K(x, x') = exp(-\gamma \|x - x'\|^2)$$

where γ is an hyperparameter that change the decision boundaries increasing the number of support vector that are considered. Overall small values of γ set decision boundaries similar to the case of linear SVM while high values set more complex boundaries because are more influenced by the support vectors.

Validation Results

Using the RBF Kernel there are two hyperparemeters to tune. In this case to choose the best couple are performed two gridsearch, one for each sampling method.As before the best model is chosen considering the best value of the F1 score but is reported also the grid with the accuracy values.

Using the undersampling method the best couple of hyperparemeters is C=1000 and $\gamma = 1$ with a value of F1 score of around 89.5 % and an accuracy of 86%.

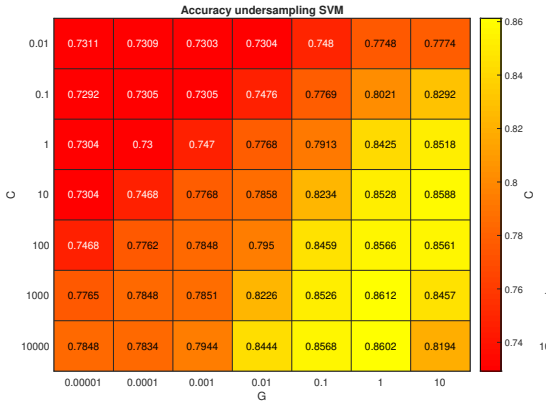


Figure 15: Accuracy

Figure 16: F1 score

Using the mix sampling method the best results are given by a different couple of hyperparameters that is C=10000 and $\gamma = 1$. In this case the results are steadily higher : the F1 score reaches around 89.7% and the accuracy 86.4%.

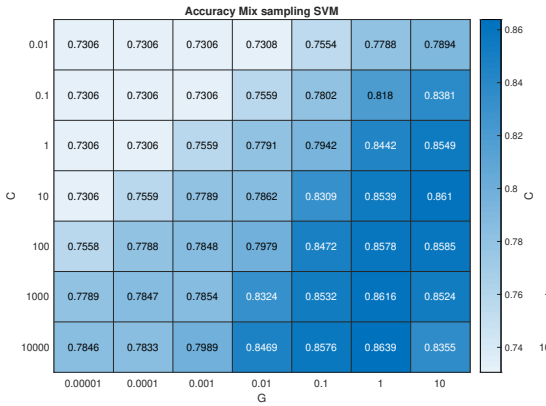


Figure 17: Accuracy

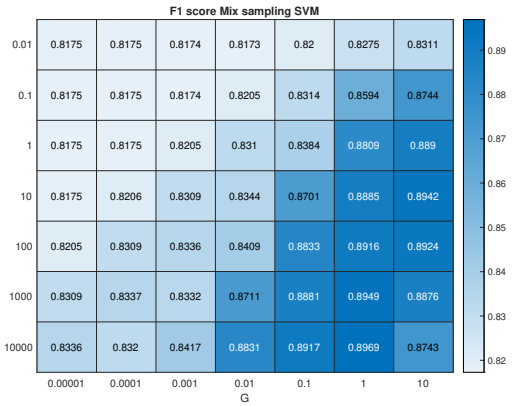


Figure 18: F1 score

In this case the mix-sampling performs better than the undersampling even if the improvements regards only few tenths.

5.4 Decision Tree

The decision tree classifier is one of the most intuitive algorithms. The main concept is to split the predictors space in non-overlapping regions. When a

new instance is classified, it is located in the respective region and take the label of the most occurring class in that region, with a sort of majority vote. To find the regions is used the binary recursive splitting on the features and is created a tree where the cutpoint and the feature to split the predictors space are chosen considering the minimization of a specific metric. In this case the metric is represented by the Giny index.

$$G = \sum_{k=1}^{K} p_{mk}(1 - p_{mk})$$

where p_{mk} represents the portion of training observations of the class k in the region m and K is the number of classes.

This index is a measure of the variance across the classes and is small when all the p_{mk} are near to 0 or 1.

This strategy can be affected by overfitting. To prevent this problem can be useful to prune back the tree to obtain a subtree. In order to optimize the dimension of the tree, the depth is considered as parameter for the validation.

Validation Results

In this case the optimization is made to optimize the dimension of the tree and are tried 4 levels of depth : 5,10,20,50.
As in the previous case the experiments are made on both the sampling methods. As can be seen by the figures 19,20 the trend of accuracy and F1 score are quite similar and it is clear that deeper trees don't performs better than smaller ones.

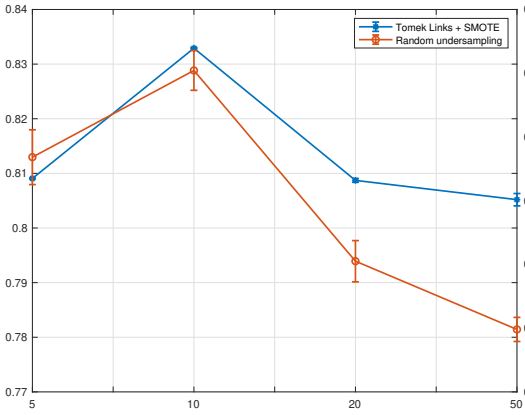


Figure 19: Accuracy

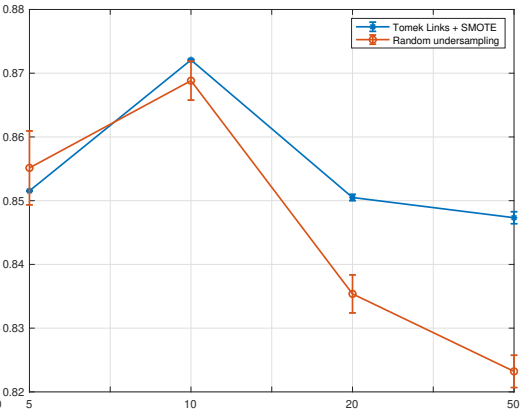


Figure 20: F1 score

5.5 Random forest

The random forest is a supervised algorithm that is based on the bagging of decision trees. The bagging method consists in creating N training set starting from the original one with boostrap. On each of this datasets a decision tree is trained using only a subset of the features. The results given by each decision tree are aggregated to obtain the overall result. More in details, each decision tree produce a response and the overall results is the most common one between the all trees.

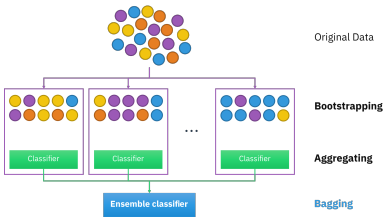


Figure 21: Bagging

The number of decision trees used in this algorithm is considered as an hyperparameter and is optimized in the next paragraph.

Validation Results

As number of decision trees are considered 7 values between 10 and 1000 and, as in the previous sections, the analysis is performed for both the sampling methods. The results obtained are reported in Figures 22, 23.

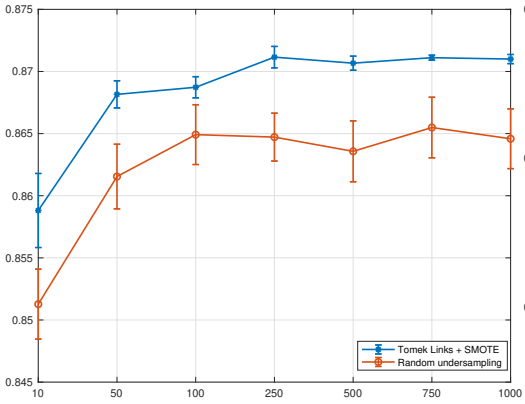


Figure 22: Accuracy

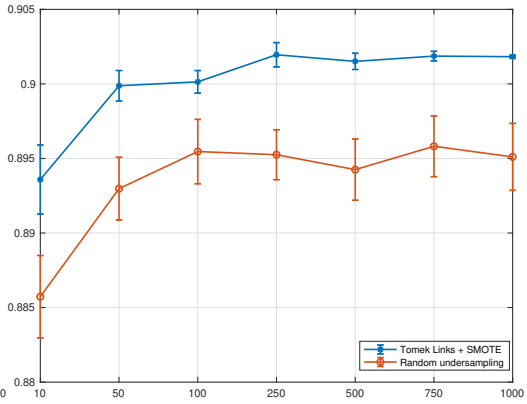


Figure 23: F1 score

Overall, the algorithm performs steadily better with the mixsampled training set and with both the sampling methods there an improvement increasing the number of trees.
At the end, with undersampling the best model contains 750 trees while with mixsampling 250.

5.6 Classifiers comparation

In this section the analyzed classifiers are tested on the test set using the best hyperparameters founded with the cross-validation (See table 1). The analysis is performed using both undersampling and mix-sampling. At the end a comparison between the classifiers is made using the ROC curve. In the curve the attention is focused for values of False positive rate under the 0.2 that is the suggested threshold.

Best validation parameters per classifiers				
	KNN	SVM	D. Tree	Random F.
Undersampling	K=15	C=1000; $\gamma = 1$	Depth=10	# Trees =750
Mixsampling	K=15	C=10000; $\gamma = 1$	Depth=10	# Trees =250

: Table1

Undersampling

Using the undersampling technique the classifiers reach high level of accuracy on the test as can be seen from the table and the confusion matrix. The best classifier seems the Random Forest with results very close to the SVM.

Undersampling test results				
	KNN	SVM	DT	RF
Accuracy	85.13	88.36	84.92	88.70
F1 score	88.71	91.22	88.62	91.30

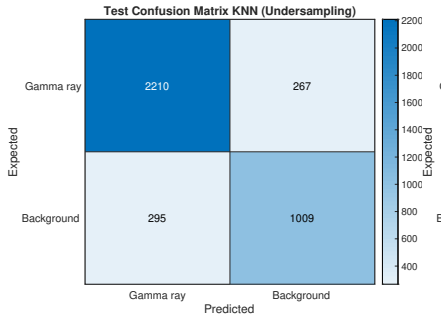


Figure 24: KNN

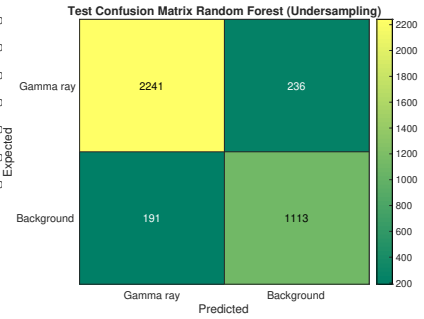


Figure 25: RF

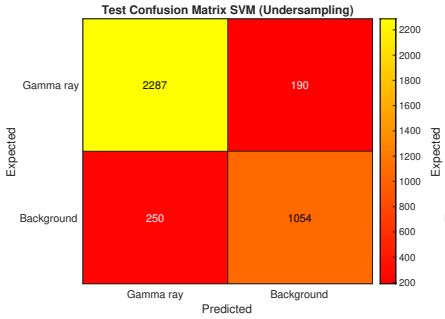


Figure 26: SVM

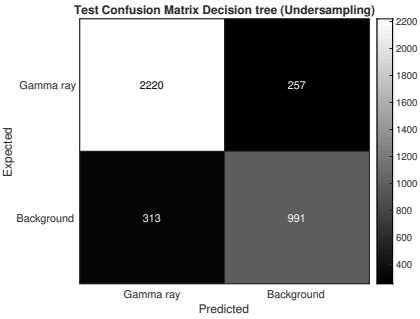


Figure 27: Decision Tree

Figure 28: Confusion Matrix on the test results for various classifiers using the undersampling technique

As expected by the results also the SVM and Random Forest curve is very close but it's clear that also in the considered range the Random forest performs better.

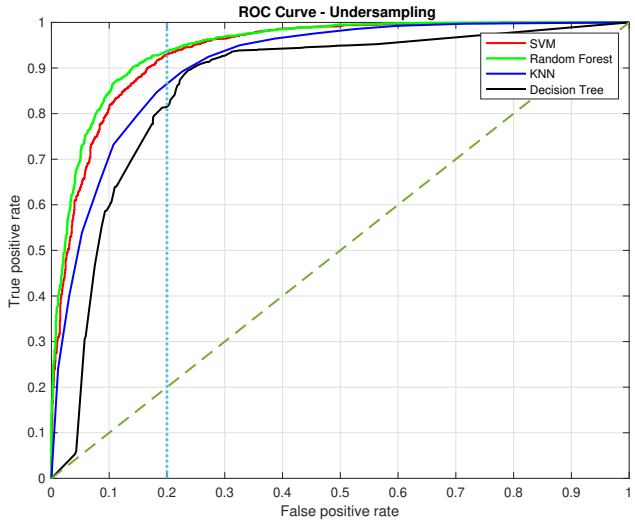


Figure 29: ROC Curve Undersampling

Mix sampling

Using the mix sampling technique the results are quite similar to the previous case. Random forest is, as before, the best classifier. However can be seen from the ROC curve in Figure 35 that in this case the SVM and Random Forest curve are more similar than in the previous case.

Mixsampling test results				
	KNN	SVM	DT	RF
Accuracy	85.32	88.36	83.81	88.78
F1 score	88.74	91.18	87.80	91.49

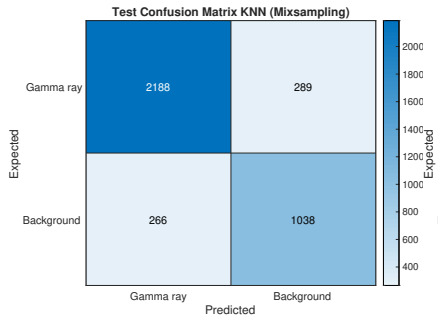


Figure 30: KNN

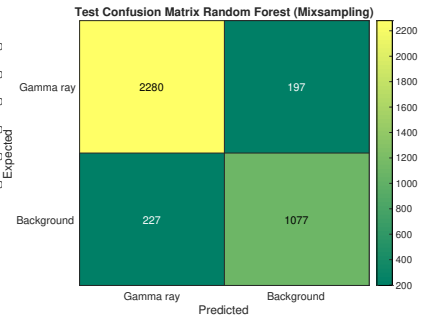


Figure 31: RF

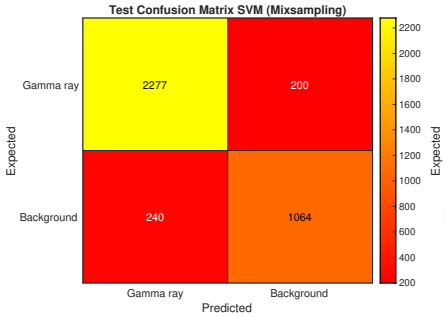


Figure 32: SVM

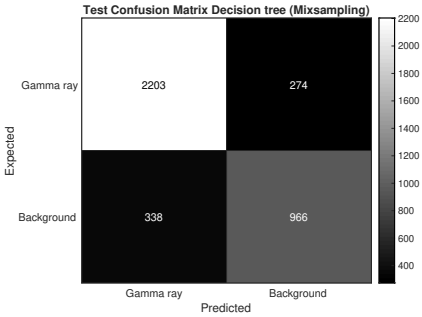


Figure 33: SVM

Figure 34: Confusion Matrix on test results for various classifiers using the mixsampling technique

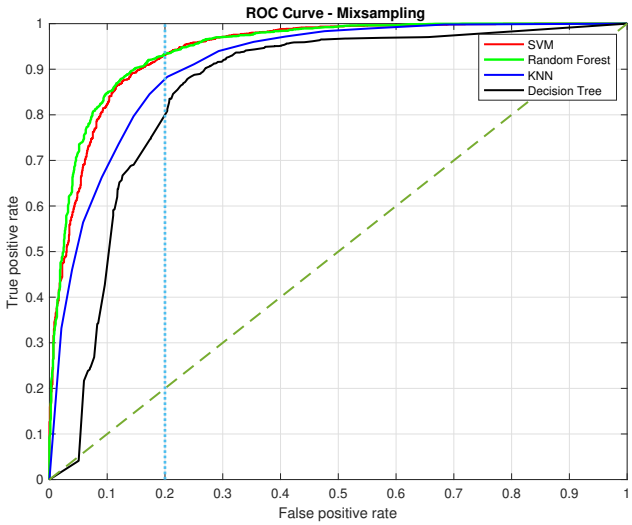


Figure 35: ROC Curve Mixsampling

6 Conclusion

Overall, from the analysis performed the Random Forest results to be the best classifier and this is clear from the F1- score but also from the analysis of the ROC Curve. Considering the sampling techniques the results are very similar even if with mixsampling are very steadily higher.

References

[1] R. K. Bock, A. Chilingarian, M. Gaug, F. Hakl, T. Hengstebeck, M. Jiřina, J. Klaschka, E. Kotrč, P. Savický, S. Towers, A. Vaiciulis, and W. Wittek, "Methods for multidimensional event classification: a case study using images from a Cherenkov gamma-ray telescope," *Nuclear Instruments and Methods in Physics Research A*, vol. 516, pp. 511–528, Jan. 2004.

[2] D. Dua and C. Graff, "UCI machine learning repository," 2017.