A COMPARATIVE STUDY OF CLASSIFIERS ON MONTE CARLO SIMULATED GAMMA AND HADRON PARTICLES IN MAGIC TELESCOPES

Divya Balasubramanian and Shruti Doshi

Abstract
In this paper, we aim to perform and critically evaluate the ability of two classification algorithms, Multilayer Perceptron (MLP) and Support Vector Machines (SVMs), to classify Monte Carlo simulated gamma and hadron particles in MAGIC telescopes. Grid search was conducted to identify the best hyperparameter combinations in each of the models for effective classification. The selected models were then compared using various performance measures. Although MLP outperforms SVMs on the unbalanced dataset, on balancing (using ADASYN) and normalising, we note that SVMs is a better classifier for this dataset.

1. Introduction

Particle physics and the distinct detection of particles in the Earth's atmosphere is an interesting area of research. When gamma rays interact with the Earth's atmosphere, they produce showers of particles that travel at high speeds [1]. Major Atmospheric Gamma Imaging Cherenkov Telescopes (MAGIC) are used to study these particles that leak into the atmosphere. The Monte Carlo simulation produces experimental data that has both gamma as well background (hadron) particles [2]. The aim of the simulation is to improve the sensitivity of the MAGIC Telescope to classify the nature of the particle appropriately. Given the large amount of background showers containing hadrons produced, there is ongoing research undertaken to find the most appropriate approach for classifying these particles accurately.

In [3], we note that Multilayer Perceptron and Support Vector Machines were used to classify the imbalanced MAGIC gamma-hadron dataset. We have experimented the same techniques to study the effect of balancing (using ADASYN) the MAGIC gamma-hadron dataset. We have undertaken grid search to optimise the models.

1.1. Multilayer Perceptron

Multilayer Perceptron (MLP) is a supervised learning technique which is most frequently used as a classifier. The architecture is composed of various simple interconnected neurons which form an input layer, one or more hidden layers and an output layer where the predictions are made. Every neuron is associated with weights, activation function, and a bias. For training the data, it uses the backpropagation technique where the predicted output is compared with the target output and an error is calculated at the output layer. This error is sent back through the network which is used to update the weights in each layer. This process is received until the error is significantly reduced or until a specific number of epochs.

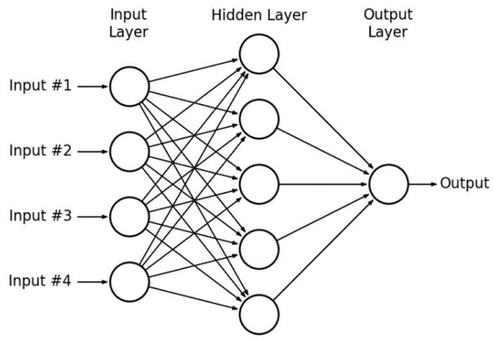

Figure 1: Model of a feed-forward Multilayer Perceptron [4]

Advantage: It can be used to model non-linear data.
Disadvantage: Its black box nature which makes it difficult to reason why the network produced a certain output.

1.2. Support Vector Machines

Support vector machines (SVMs) are a supervised machine learning algorithm popularly utilised for classification models. SVMs aim to find a hypothesized third dimension, called a hyperplane, that classifies the data into their respective classes in an N-dimensional space (where N is the number of features). For non-linearly separable data, the dot product of the input data points are projected to a higher dimensional space which is then divided by a hyperplane. This dot product is called the kernel function. The data points closest to the hyperplane, on either side of the decision boundary, are called the support vectors. The distance between the support vectors and the hyperplane defines the margin of the model. The larger the margin distance, the better confidence of the classification.

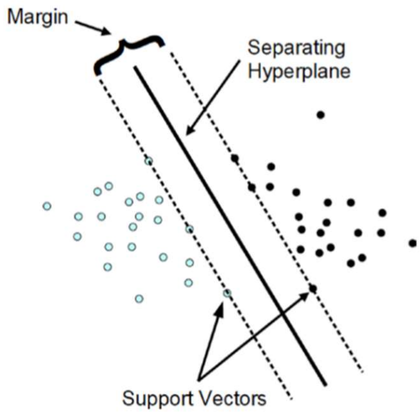

Figure 2: Model of Support Vector Machines [5]

Advantage: Its ability to produce a unique solution when the optimal hyperparameters are determined. This forms a fundamental difference from MLP which produces varied outputs based on the local minima.
Disadvantage: It is computationally expensive due to its slow nature [6].

2. Hypothesis

As Neural Networks are universal functional approximators [7], our null hypothesis is that MLP should outperform SVMs for this dataset. However, we note [8] concludes that SVMs outperform neural networks on a balanced dataset.

3. Dataset

The dataset used for this study is obtained from the UCI repository [9]. The classes are gamma rays (signal) and hadron showers (background) produced by Monte Carlo simulation. Each sample is described by 10 features. There is class imbalance with 12,332 gamma event samples and 6,688 hadron event samples. In the following report, Gamma is considered as Class 1 and Hadron as Class 2. Table 1 displays the summary statistics of the dataset.

	Gamma					Hadron				
	mean	std-dev	min	max	skew	mean	std-dev	min	max	skew
fLength	43.65	26.17	12.19	272.06	1.24	70.94	57.95	4.28	334.18	1.28
fWidth	18.59	9.03	0.00	176.34	2.48	28.80	27.19	0.00	256.38	2.12
fSize	2.78	0.46	2.00	5.01	0.82	2.90	0.48	1.94	5.32	0.98
fConc	0.38	0.18	0.01	0.89	0.57	0.37	0.19	0.01	0.89	0.37
fConc1	0.22	0.11	0.01	0.68	0.74	0.21	0.12	0.00	0.64	0.61
fAsym	3.24	39.63	-349.76	219.90	-0.92	-18.29	82.30	-457.92	575.24	-0.53
fM3Long	17.81	33.94	-198.87	215.89	0.14	-2.85	70.69	-331.78	238.32	-0.79
fM3Trans	0.19	13.55	-91.35	101.39	0.08	0.36	29.92	-205.89	179.85	0.09
fAlpha	18.78	21.48	0.00	90.00	1.55	43.99	25.98	0.01	90.00	0.07
fDist	190.23	70.30	5.75	450.40	0.19	200.43	81.87	1.28	495.56	0.21

Table 1: Summary statistics of the dataset

3.1. Initial Data Analysis

The features of the dataset were not in the same range due to which normalisation has been performed. It is observed that by performing normalisation the accuracy of the SVM model could increase [10]. To handle the class imbalance, ADASYN (extension of SMOTE) is applied which uses a synthetic data generation technique to increase the number of samples in the minority class. As a result, the number of samples in class ‘h’ (hadron) increases up to 11,954. In MATLAB, the neural network architecture requires the input to be one-hot encoded which was performed before feeding the dataset to the classifier. Figure 3 shows the histograms of features of the gamma class(blue) and hadron class(orange). It can be clearly observed that there are no features that distinctively differ for both the classes.

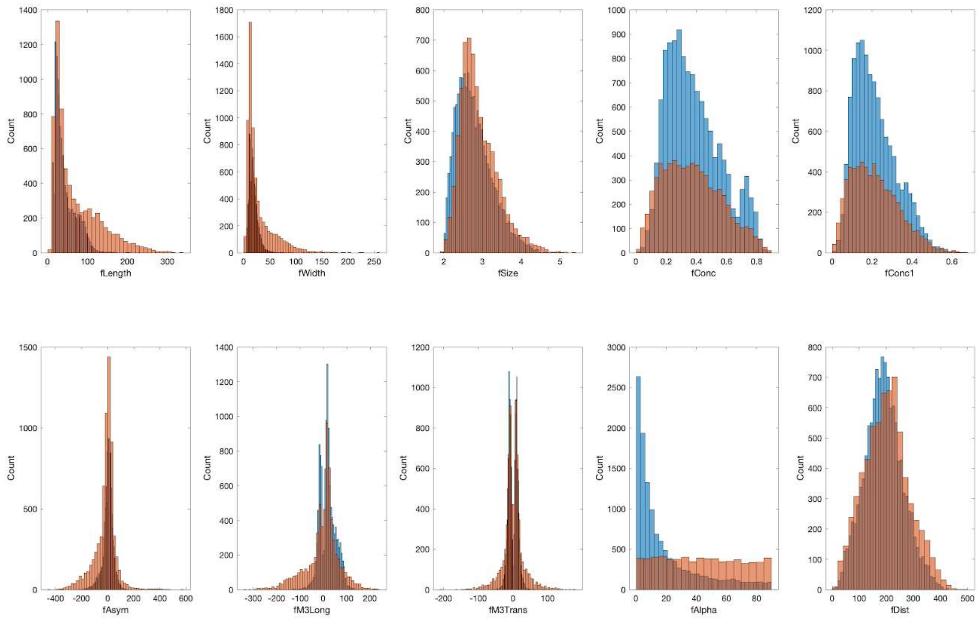

Figure 3: Histograms of features of gamma class and hadron class

4. Methodology

The original dataset is divided into training and testing. 70% of the dataset is used for training and validation while the remaining 30% is used for testing the models. Hyper parameter tuning is done to perform a grid search of the models. The best model was then chosen and applied on the test data. A 10-fold cross validation was used while performing the grid search. Average of the validation accuracies over the 10-folds of each hyperparameter combination was calculated and based on this, the best model was selected.

Assumption: All hyperparameters were chosen based on few trial runs on MATLAB to narrow down the range of values for grid search.

4.1. Architecture and Parameters used for the MLP

The hyperparameters chosen for tuning our model included number of neurons in the hidden layer, learning rate and the momentum. We considered one hidden layer for our model to avoid overfitting. The number of neurons in that layer were changed to 5, 10, 20 and 50. The learning rate decides rate by which the weights in the network are changed. When the learning rate value is high, rapid changes are made to the weights per update, whereas when it is low, the weights are changed more gradually. The learning rate values considered were 0.003, 0.01 and 0.03. Momentum can be used to prevent the system from getting stuck in a local minima. The values considered for momentum were 0.3, 0.6 and 1.0.

4.2. Architecture and Parameters used for the SVMs

The hyperparameters chosen to tune the SVMs model include box constraint and kernel function. Non-linearly separable data points need kernel tricks to identify the most suitable kernel function for the hyperplane. The SVMs function in MATLAB supports Linear, Gaussian and Polynomial kernel functions, all of which were tested on this dataset to select the most suitable parameters. Within the polynomial kernel function, polynomial orders of 2, 3 and 4 which are supported by MATLAB fitcsvm [11] were tested. The box constraint adds a cost for misclassification wherein, the higher the box constraint, the higher the cost of misclassification thereby leading to better separation of classes. However, large values of box constraint lead to high computational time. Hence a geometric sequence of box constraints including 0.3,0.4, 0.5 and 0.6 were chosen for grid search.

5. Results, Findings & Evaluation

5.1. Model Selection

Using grid search, the hyperparameters were tuned to identify the best model in each case. K-fold cross validation was used to ensure the model is adequately trained and the mean accuracy was obtained. In MLP, since the initial weight assignment is random, this greatly affects the performance of that model. Upon running the experiment again, using the same grid-search, we note that the accuracies of the models change every time. SVMs are highly sensitive to the combination of hyperparameters. The box constraint particularly plays an important role in defining the decision boundary as an increase in box constraint considers fewer support vectors for the classification.

Table 2 lists out the average accuracy obtained from each hyperparameter combination tested on both MLP and SVMs. The accuracies are distinguished with colors to highlight the combinations of hyperparameters that are performing better. The parameters highlighted in yellow for both the classifiers formed the best models for this dataset.

MLP

Momentum	Learning_rat	Num_neurons	AccuracyValue
0.3	0.003	5	71.059
0.3	0.003	10	74.776
0.3	0.003	20	71.571
0.3	0.003	50	71.576
0.3	0.01	5	74.753
0.3	0.01	10	75.894
0.3	0.01	20	74.576
0.3	0.01	50	64.847
0.3	0.03	5	74.653
0.3	0.03	10	73.941
0.3	0.03	20	74.971
0.3	0.03	50	70.218
0.6	0.003	5	74.706
0.6	0.003	10	74.594
0.6	0.003	20	73.118
0.6	0.003	50	67.759
0.6	0.01	5	73.288
0.6	0.01	10	72.035
0.6	0.01	20	73.418
0.6	0.01	50	66.176
0.6	0.03	5	74.724
0.6	0.03	10	73.741
0.6	0.03	20	75.012
0.6	0.03	50	68.841
1	0.003	5	72.782
1	0.003	10	75.512
1	0.003	20	71.206
1	0.003	50	60.924
1	0.01	5	74.106
1	0.01	10	75.188
1	0.01	20	72.353
1	0.01	50	62.024
1	0.03	5	73.224
1	0.03	10	73.871
1	0.03	20	73.706
1	0.03	50	70.559

SVM

Kernel	Box Constrai	Poly Order	Loss	Accuracy
Gaussian	0.3	0	0.20404575	81
Gaussian	0.4	0	0.20141176	79.3529412
Gaussian	0.5	0	0.19983007	79.7058824
Gaussian	0.6	0	0.19829412	80.3529412
Linear	0.3	0	0.27961438	72.5882353
Linear	0.4	0	0.27973203	74.8823529
Linear	0.5	0	0.27930719	71.0588235
Linear	0.6	0	0.27930065	71.7058824
Polynomial	0.3	2	0.18786275	79.1764706
Polynomial	0.3	3	0.18630065	80.5882353
Polynomial	0.3	4	0.18536601	81.5882353
Polynomial	0.4	2	0.18693464	80.2941176
Polynomial	0.4	3	0.18529412	80.4705882
Polynomial	0.4	4	0.18444444	81.1764706
Polynomial	0.5	2	0.18626797	78.8235294
Polynomial	0.5	3	0.18477778	80.5294118
Polynomial	0.5	4	0.18409804	80.5294118
Polynomial	0.6	2	0.18586928	80.7058824
Polynomial	0.6	3	0.18438562	81.8235294
Polynomial	0.6	4	0.18381046	82.2941176

Table 2: Grid Search for Hyperparameter optimisation for MLP and SVM

5.2. Analysis and Critical Evaluation of Results

Both MLP and SVMs were first run on the imbalanced data extracted from the UCI repository to test the initial performance scores obtained from the confusion matrix. The accuracy is the percentage ratio of correctly predicted observations to the total observations. Precision refers to the percentage of the classified results which are rightly classified whereas recall gives a percentage of total relevant results correctly classified. F1 score provides a weighted average of precision and recall.

Table 3 depicts the results obtained. It is seen that for unbalanced data MLP performs better with an accuracy of 82.16% compared to SVMs that has an accuracy of 72.54%.

Model	Precision	Recall	F1 Score	Accuracy
SVM	0.7961	0.7034	0.7469	72.54
MLP	0.9199	0.8195	0.8668	82.16

Table 3: Performance measures of models on imbalanced dataset

However, upon balancing the data using ADASYN, the best models with tuned hyperparameters were executed on the testing data. The confusion matrices were plotted (Figure 4a and Figure 4b) giving us performance scores as seen in Table 4. We see that, in this case, SVMs have a better accuracy of 81.48% as compared to MLP with 71.88%. From the confusion matrices, we infer that MLP mis-classifies 35% more than SVM.

Model	Precision	Recall	F1 Score	Accuracy
SVM	0.86326	0.78858	0.82423	81.482
MLP	0.71015	0.7252	0.7176	71.887

Table 4: Performance measures of models on balanced and normalised test set

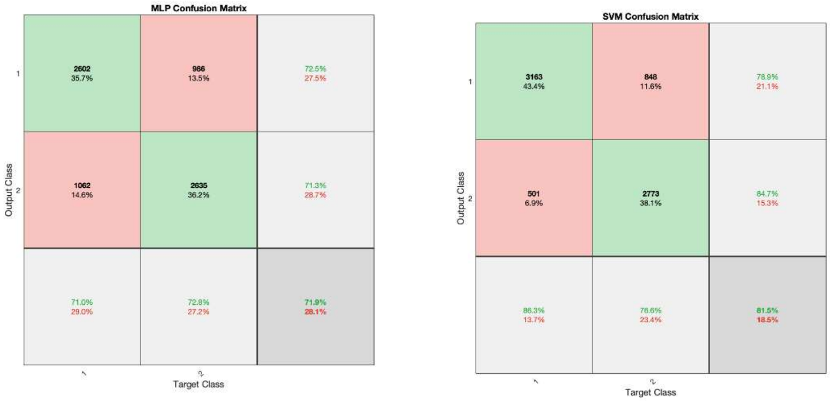

Figure 4a, 4b: Confusion Matrix for MLP and SVMs obtained after applying the best model on the test set

A ROC (Receiver Operator Curve) was plotted to enhance the analysis of our experiment. It is useful to display the relationship between the sensitivity (True Positive Rate) and specificity (True Negative Rate). ROC does not depend on class distribution and is hence one of the most appropriate measures to analyse the performance of the algorithms. Also, AUC (Area Under the ROC Curve) is a good indicator of the degree of separability between the classes. Higher AUC signifies that the model performance is good. It can be observed from the ROC generated from the testing phase (Figure 5) that SVM performs better than MLP for both the classes.

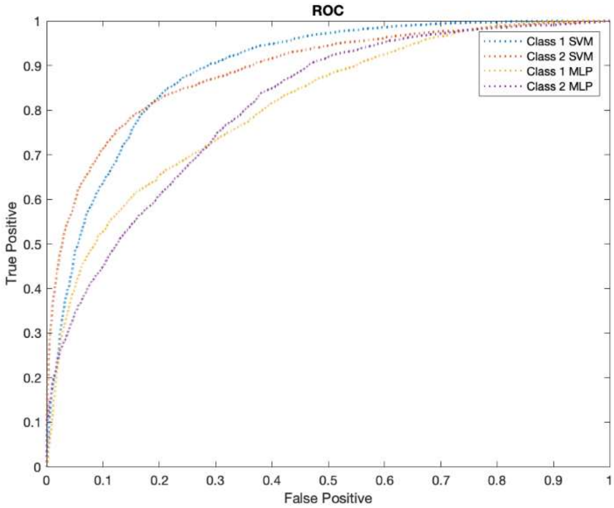

Figure 5: ROC of the performance of MLP and SVMs on the testing set held out in the beginning of the experiment

We observed that SVM took 1775.87 seconds whereas MLP took 441.2 seconds to perform the classification. This proves that SVM is computationally expensive as for every instance of unseen data, the algorithm re-calculates the distance to the current hyperplane [12].

In a similar experiment conducted in [13] it is noticed that SVM performs better than MLP when the models are run on larger datasets. However, from [5, 14] we note that balancing the classes in the data plays a significant role in performance of the models to allow balanced learning. There are two main reasons SVM does not perform as well when there is class imbalance. These are weakness of the soft margin optimization problem and the imbalanced support vector ratio [14]. The former focuses on the soft margin ‘c’, which in our case is the box constraint, which causes a bias in the position of the hyperplane when there are higher number of samples in the majority class. This skewed position of the hyperplane causes prediction of more false negatives thereby affecting the overall accuracy of the model. The support vector imbalance focuses on the sparse nature of the minority class support vectors which causes an increase in likelihood of the classification of a test sample to be false positive. The minority class in our imbalanced dataset was only 35% of the total samples which makes it evident as to why SVMs were not as efficient before balancing.

6. Conclusion

Our experiments aim to classify gamma and hadron particles generated by Monte Carlo simulations. It is critical to identify a model with high accuracy as classification of a hadron as a gamma ray can prove dangerous and can affect the sensitivity of MAGIC telescopes.

Based on our observations and evidence from literature, we conclude that for this dataset, SVMs are a better classifier than MLP thus rejecting our null hypothesis. We note that data imbalance plays a vital role in the efficiency of the models. We observed that all performance measures as well as ROC were consistent in identifying SVMs as a better classifier in our experiments. Although, it must be mentioned that the computational cost for SVMs associated with the time required to execute the models are relatively higher than MLP.

For future work, we would be interested in performing a deep-dive to understand the effect of introducing more data points through other sampling techniques such as MWMOTE used in [15]. It would also be interesting to investigate the cost associated with the SVMs model and compare it with the associative cost of MLP.

7. References

[1] Cherenkov Telescope Array. (2019). How CTA Works - Cherenkov Telescope Array. [online] Available at: https://www.cta-observatory.org/about/how-cta-works/ [Accessed 30 Mar. 2019].

[2] Majumdar, P., Moralejo, A., Bigongiari, C., Blanch, O. and Sobczynska, D., 2005, August. Monte Carlo simulation for the MAGIC telescope. In Proc. 29th Int. Cosmic Ray Conf (pp. 41-44).

[3] Boinee, P., De Angelis, A. and Foresti, G.L., 2005. Ensembling Classifiers-An application to image data classification from Cherenkov telescope experiment. In IEC (Prague) (pp. 394-398).

[4] Shaker, A., Ali, A.H., Hamed, M., Saleh, T. and Emad, A., 2015. ASSESSMENT OF ARTIFICIAL NEURAL NETWORK FOR BATHYMETRY ESTIMATION USING HIGH RESOLUTION SATELLITE IMAGERY IN SHALLOW LAKES: CASE STUDY EL BURULLUS LAKE.

[5] En.proft.me. (2019). How-to simulate Support Vector Machine (SVM) in R | en.proft.me. [online] Available at: https://en.proft.me/2014/04/22/how-simulate-support-vector-machine-svm-r/ [Accessed 31 Mar. 2019].

[6] Karamizadeh, S., Abdullah, S.M., Halimi, M., Shayan, J. and javad Rajabi, M., 2014, September. Advantage and drawback of support vector machine functionality. In 2014 International Conference on Computer, Communications, and Control Technology (I4CT) (pp. 63-65). IEEE.

[7] En.wikipedia.org. (2019). Universal approximation theorem. [online] Available at: https://en.wikipedia.org/wiki/Universal_approximation_theorem [Accessed 31 Mar. 2019].

[8] Strathprints.strath.ac.uk. (2019). [online] Available at: https://strathprints.strath.ac.uk/48403/1/ANN_SVM_v4.pdf [Accessed 31 Mar. 2019].

[9] Archive.ics.uci.edu. (2019). UCI Machine Learning Repository: MAGIC Gamma Telescope Data Set. [online] Available at: https://archive.ics.uci.edu/ml/datasets/magic+gamma+telescope [Accessed 31 Mar. 2019].

[10] KumarSingh, B., Verma, K. and S. Thoke, A. (2015). Investigations on Impact of Feature Normalization Techniques on Classifier's Performance in Breast Tumor Classification. International Journal of Computer Applications, 116(19), pp.11-15.

[11] Uk.mathworks.com. (2019). Train support vector machine (SVM) classifier for one-class and binaryclassification - MATLAB fitcsvm- MathWorks United Kingdom. [online] Available at: https://uk.mathworks.com/help/stats/fitcsvm.html [Accessed 31 Mar. 2019].

[12] He, H. & Garcia, E.A. 2009, "Learning from Imbalanced Data", IEEE Transactions on Knowledge and Data Engineering, vol. 21, no. 9, pp. 1263-1284.

[13] Barabino, N., Pallavicini, M., Petrolini, A., Pontil, M. and Verri, A. (1999). Support vector machines vs multi-layer perceptrons in particle identification. In: ESANN'1999 proceedings - European Symposium on Artificial Neural Networks. [online] Bruges, pp.pp. 257-262. Available at: https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es1999-463.pdf [Accessed 31 Mar. 2019].

[14] Batuwita, R and Palade,V (2012) ‘Class Imbalance Learning Methods for Support Vector Machines’, in He, H and Ma, Y (ed.) Imbalanced Learning: Foundations, Algorithms, and Applications. Published Online: John Wiley & Sons, Inc., Chapter 6 pp.1-20.

[15] Pattanayak S.S., Rout M. (2018) Experimental Comparison of Sampling Techniques for Imbalanced Datasets Using Various Classification Models. In: Saeed K., Chaki N., Pati B., Bakshi S., Mohapatra D. (eds) Progress in Advanced Computing and Intelligent Engineering. Advances in Intelligent Systems and Computing, Singapore: Springer, vol 564 pp.13-22.