Fast and Accurate Gamma-Ray/Hadronic Particle Shower Classification Using Machine Learning

Leonardo Valli

9/28/24

Inspirit AI Research Mentorship

Abstract

Correct and efficient prediction of atmospheric gamma-ray particle showers is important because the study of these initiating particles can lead us to discoveries about the parts of the universe from which they originated. Previous research has been done with datasets similar to the one we chose, all of which go deep into the optimization of machine learning methods to solve this classification problem. However, we have yet to see researchers try to simplify their model in order to maintain high model performance while minimizing the number of features used. Because of this, we used dimensionality reduction methods like Pearson Correlation and Principal Component Analysis on sklearn classification models like Random Forests and Support Vector Classifiers to try and solve this issue. It was found that with a Pearson Correlation Coefficient cut-off of 0.01, model accuracy was increased from 87.51% to 87.83% while reducing the number of features used by two, meaning our goal was accomplished. In addition, a stacking ensemble yielded an accuracy of 88.04% with fast predicting speed.

1. Introduction

Astrophysics is a continually growing field, as there is much to discover outside of the boundaries of our planet. One key avenue for learning about the makeup of the universe is via the study of particles that were created somewhere else in the universe and traveled to Earth. The goal of our research was to build a simple scikit-learn machine learning classification model to differentiate between gamma and hadronic particle showers detected by the Major Atmospheric Gamma Imaging Cherenkov (MAGIC) Telescope. Specifically, the task was to maximize the percentage of gamma-ray-induced showers properly classified as such, while minimizing the number of hadronic showers classified as gamma-ray showers, while also taking into account the amount of time and number of features of the dataset used to do so.

Gamma-ray showers are particle showers initiated by gamma-rays hitting particles in Earth's atmosphere, while hadronic showers are showers of particles including hadrons that are initiated by the collision of cosmic rays with particles in the atmosphere. While in the case of both gamma and hadronic showers it is true that there is a ray colliding with a particle in Earth's atmosphere, the difference between these two events is the variety of particles involved in the shower. In the case of gamma ray showers, the showers are made up of mostly electrons and positrons. However, due to the more complex makeup of cosmic rays, showers initiated by them involve different kinds of particles, including protons, nuclei, and hadrons [1]. In both scenarios, as the particles descend, they are moving at almost the speed of light, resulting in the creation of Cherenkov radiation. The MAGIC telescope uses a parabolic mirror to reflect this radiation into a focus: a camera [2]. Features of the resulting images taken by this camera, such as distribution and intensity of the Cherenkov radiation that is imaged, can be used to extract conclusions about the originating particle, such as its direction.

Cosmic ray showers are much more common than gamma ray showers, but gamma rays are a lot more important for astrophysics research, because they can be better used for research into where they originate from, environments such as black holes, neutron stars, and more [3]. As both situations are detected by Cherenkov Gamma Telescopes such as the MAGIC Telescope, that is why it is important to distinguish between events detected by telescopes such as these, and do it efficiently as well [4]. That makes this issue a classification problem, using the numerical data drawn from features of images taken by the MAGIC telescope. The models we trained were meant to be able to take these numerical inputs and use as few of these features as possible in order to correctly predict gamma showers with as high of an accuracy as possible.

2. Background

Researchers such as Bock et al. [5] and Albert et al. [6] have worked with very similar datasets to those that we used. Bock et al. uses machine learning algorithms such as Random Forests, Kernel Probability Density Estimation, and more. Albert et al. also uses Random Forests, but goes deeper into optimization and using different versions of Random Forest to maximize performance. Their work contributed greatly to the workpool with the MAGIC Telescope dataset, exhibiting numerous in-depth methods of classification that were useful for gathering insight into which methods worked best for that specific dataset.

Because of the popularity of the Random Forest method in previous work into gamma/hadron shower classification, we felt it important to implement our own version of Random Forests as well. However, we decided to focus on keeping our machine learning models as simple as possible, for the purpose of achieving the highest accuracy possible while keeping the model simple.

In addition, Luo et al. [7], who worked with a slightly different dataset, implemented an automatic feature selection algorithm in order to use the most effective features of their dataset to maximize accuracy while minimizing the number of features used to do so. This work was also inspirational in our work and our methods of attempting to simplify the model without hurting model performance.

3. Dataset

For this project, we used the MAGIC Telescope Dataset from 2004. The dataset contains data from 19020 instances of particle shower images registered by the MAGIC Cherenkov Gamma Telescope. The dataset's features are entirely numerical and continuous, with a qualitative binary class that can either be 'g' for gamma, or 'h' for hadron. Each of the particle showers described are caused by a particle, either a gamma ray or a cosmic ray, hitting Earth's atmosphere and initiating a chain reaction of particle launches and separations that eventually lose all their energy and die out as they descend through the atmosphere.

The class 'g' signifies particle showers detected that were caused by gamma rays hitting Earth's atmosphere. The class 'h' represents particle showers initiated by cosmic rays hitting particles in Earth's atmosphere, splitting into other secondary particles such as hadrons. Features of the dataset are features of the images of Cherenkov radiation originating from the showers taken by the MAGIC telescope.

The features in the MAGIC Telescope Dataset are as follows:

Attribute	**Description**
Index	position of instance in ordered dataset
fLength	major axis of ellipse [mm]
fWidth	minor axis of ellipse [mm]
fSize	10-log of sum of content of all pixels [in #phot]
fConc	ratio of sum of two highest pixels over fSize [ratio]
fConc1	ratio of highest pixel over fSize [ratio]
fAsym	distance from highest pixel to center, projected onto major axis [mm]
fM3Long	3rd root of third moment along major axis [mm]
fM3Trans	3rd root of third moment along minor axis [mm]
fAlpha	angle of major axis with vector to origin [deg]
fDist	distance from origin to center of ellipse [mm]

Table 1: MAGIC Telescope Dataset Feature Descriptions

An initial review was done of the attributes in this dataset to predict which ones would yield the most predicting power for our class attribute. This was done by plotting each attribute against each of the others, and separating instances of different classes by color. Then, scatter plots that showed noticeable separation between the distributions of paired attributes were kept, and the attributes that made them up were noted as attributes that may be useful in the separation of gamma and hadron predictions. As shown below, the attribute fSize showed signs of having high predicting power, as well as features like fDist, fLength, fWidth, and fAlpha.

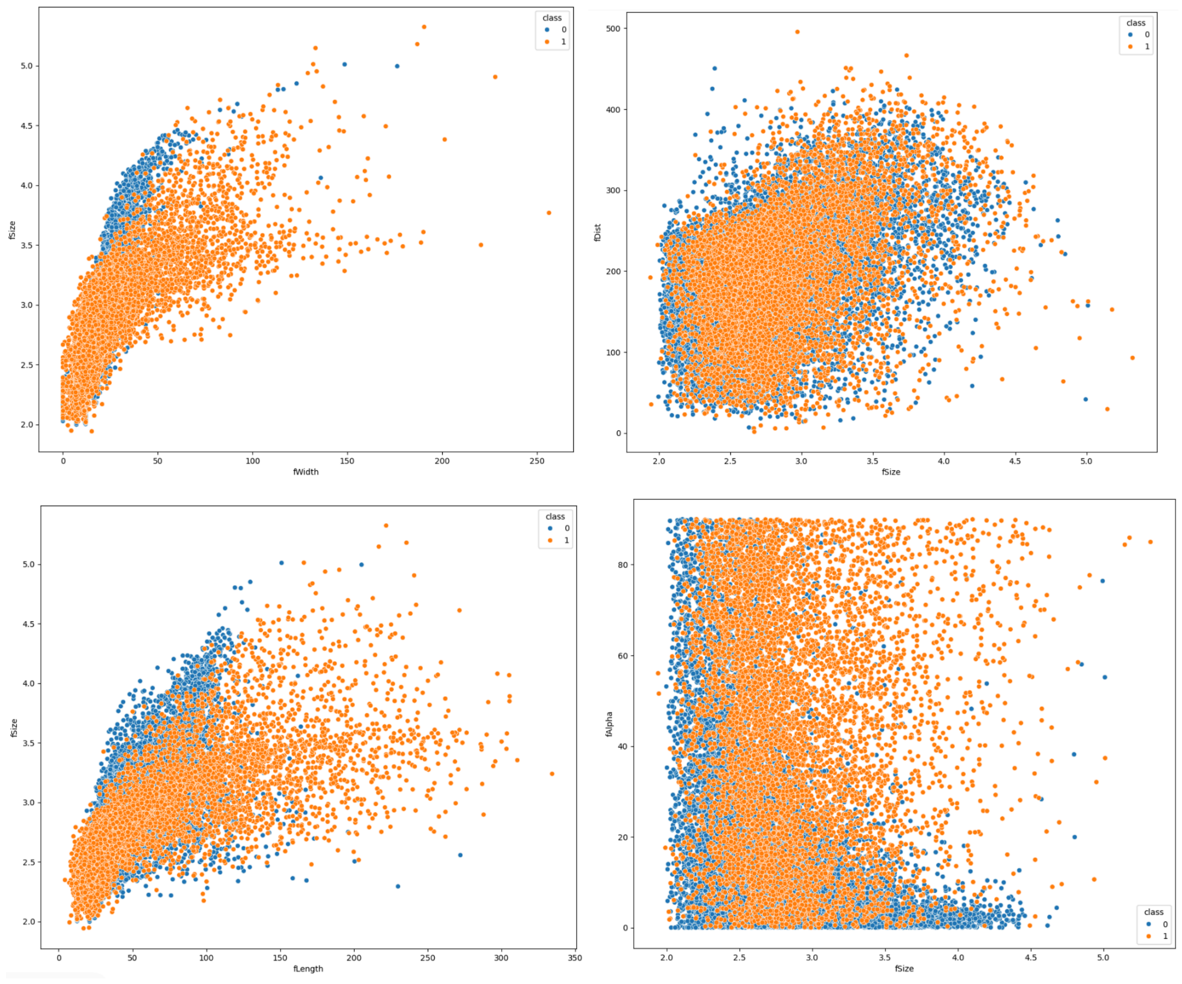

Figure 1: Dataset features plotted against each other, gamma (blue) and hadron (orange). fWidth vs. fSize (top left), fSize vs. fDist (top right), fLength vs. fSize (bottom left), fSize vs. fAlpha (bottom right)

In addition, histograms like the one shown below also gave the impression that fAlpha would be a good class separator due to its difference in distribution between the gamma class (in green on the left) and the hadron class (in yellow on the right).

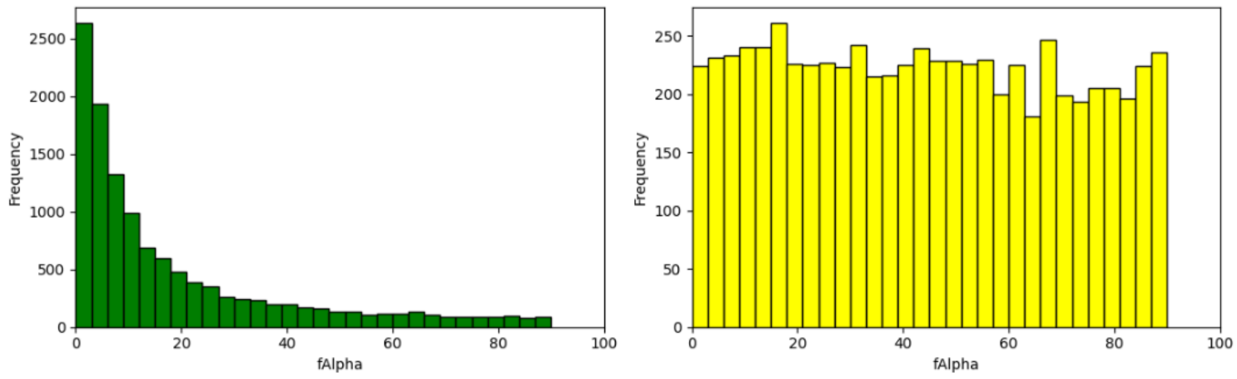

Figure 2: fAlpha value distribution histograms in Gamma class (left) vs. Hadron class (right)

The data were preprocessed by removing the Index attribute due to its uselessness in predicting the class. Further attribute selection was performed using the Pearson correlation coefficient ranking method described in the Methodology section, but most of our models were trained on all ten attributes. Data were randomly separated using the sklearn train_test_split() method with an 80-20% splitting ratio. random_state=42 was used to ensure reproducibility. The resulting training data had 15216 instances, while the testing set had 3804 instances.

4. Methodology / Models

The first goal was to find a simple sklearn machine learning model that would perform well on the data with little added effort for excessive optimization. We started by training five different base sklearn classification models on the train-test divided data mentioned before. The classification models tested were a Logistic Regression model, Ridge Classifier, Random Forest Classifier, Decision Tree Classifier, and Support Vector Classifier. Basic parameters were used, simply to get a sense of which models would perform the best without any modifications made. Accuracy was calculated to measure performance of each model on the testing set after training, as well as other performance metrics such as the precision, recall, and F1-score of the model for each class.

	Models				
	LogisticRegression()	RidgeClassifier()	RandomForestClassifier(max_depth=2, random_state=0)	DecisionTreeClassifier(random_state=0)	SVC(gamma='auto')
Accuracy	78.63%	78.71%	75.16%	81.13%	87.12%
Precision (Gamma class)	80.04%	79.28%	72.67%	85.58%	85.90%
Precision (Hadron class)	74.98%	77.08%	93.09%	73.08%	90.28%
Recall (Gamma class)	89.19%	90.81%	98.70%	85.16%	95.81%
Recall	59.30%	56.55%	32.07%	73.74%	71.21%

(Hadron class)					
F1-Score (Gamma class)	84.37%	84.65%	83.71%	85.37%	90.58%
F1-Score (Hadron class)	66.22%	65.24%	47.70%	73.41%	79.62%

Table 2: Performance metrics of base sklearn models

The next step we took to try and maximize model performance was hyperparameter tuning. Parameters of each of the five models tested were selected based on their likelihood of affecting model performance, and various possible values were selected for testing from each of these parameters. Each combination of parameter values was tested, and the combination of parameters that led to the highest accuracy on the testing set was recorded. The number of parameters selected for tuning for each model depended on the base model's speed, along with the loss of speed caused by variations in parameter values. Fewer parameters were selected for models that generally took longer to train, in order to minimize time taken to tune the parameters.

Parameters selected for hyperparameter tuning for the Logistic Regression model were penalty, tol, C, fit_intercept, intercept_scaling, class_weight, solver, lbfgs, max_iter, multi_class, verbose, warm_start, and n_jobs. Parameters selected for the Ridge Classifier model were alpha, solver, and fit_intercept. Parameters selected for the Random Forest Classifier model were max_depth, min_samples_split, and min_samples_leaf. Parameters selected for the Decision Tree Classifier model were max_depth, min_samples_split, min_samples_leaf, and max_features. Parameters selected for the Support Vector Classifier model were C, kernel, gamma, and degree. Results of this hyperparameter tuning were as follows (parameter values not shown can be assumed to be the default sklearn values):

	Models				
	LogisticRegression()	RidgeClassifier()	RandomForestClassifier()	DecisionTreeClassifier()	SVC()
Accuracy	79.18%	78.79%	88.22%	84.75%	87.62%
Parameters	{tol= 0.001, c= 0.1, class_weigh=	{alpha= 0.1}	{min_samples_split= 5, min_samples	{max_depth= 10}	{C= 10}

	balanced, max_iter= 200, multi_class= auto}		_leaf= 2}		

Table 3: Highest accuracies of base sklearn models during hyperparameter tuning, with parameter values to achieve them.

Overall, it can be seen that the Random Forest Classifier and Support Vector Classifiers performed the best out of the five, so from here on, all other models used were built off of these two sklearn classifiers specifically. Optimal parameter values for the Random Forest Classifier varied from run to run due to the randomness present in the Random Forest algorithm. Because of this, while the highest-performing parameters on this run were {‘max_depth’: None, 'min_samples_split': 5, 'min_samples_leaf': 2}, for the rest of this paper, “optimized parameters” will refer to the Random Forest parameter values {‘max_depth’: 20, 'min_samples_split': 2, ‘min_samples_leaf’: 2}, as these values resulted in fairly consistent high performance throughout trials.

Random Forests

Random Forest is an ensemble learning method that trains multiple individual decision trees on different subsets of the training data using bagging [6]. Decision Trees make predictions by making splits in the data based on rules formed from the values of features. Every branch off in the tree is a separate rule that is splitting the data, and the leaves of the tree are class labels. Random Forest combines predictions of these Decision Trees to make a more well-informed decision on what the class of the instance being tested on is.

Support Vector Classifiers

The goal of a Support Vector Classifier is to find the hyperplane that best separates the instances of the two classes in the feature space [8]. Moreover, the aim is to maximize the distance from this plane to the nearest instances of each class on either side of it, called support vectors. The generic equation for this hyperplane of separation is $w^Tx+b=0$, where w is a vector of weights for each feature, x is a vector of the input features, and b is a value used to shift the plane in the feature space. If $w^Tx+b>0$ for a given input vector, the instance will be classified as one class. Otherwise, it will be classified as the other class.

Ensembling

The next method we attempted to use to increase model performance was ensembling. Ensembling is a method that uses multiple classification models and learns how to either combine the predictions from each model, or know which model is better at predicting certain types of instances [9]. The first ensemble model used was a Bagging Classifier using the hyperparameter-optimized Support Vector Classifier as its base model that it would build multiple of. The second ensemble model used was a Stacking Classifier, using the hyperparameter-optimized Support Vector Classifier and Random Forest Classifiers as base models.

Attribute Selection/Dimensionality Reduction

The main purpose of attempting different attribute selection algorithms was to determine if there was a way to maintain high performance while minimizing the number of attributes used to do so. This is important, because a reduction in number of attributes would result in lowered model complexity, lower training and testing times, and lower storage requirements assuming vast amounts of similar data to the data we have were acquired.

Pearson Correlation Coefficient Ranking

The first method of attribute selection used was ranking based on the Pearson Correlation Coefficient [10] of each feature in the dataset. Pearson correlation is a metric that tells the correlation between each feature and the class, and therefore the predicting power that the particular feature has. Values of this coefficient range from -1 to 1, with values closer to -1 or 1 signifying high correlation, and values closer to 0 signifying low feature correlation. The results of the Pearson Correlation Coefficient calculations were as follows:

Attribute	**Pearson Correlation Coefficient**
fAlpha	0.46098
fLength	0.30757
fWidth	0.2656
fSize	0.11779
fDist	0.0652
fM3Trans	0.00384
fConc1	-0.0048
fConc	-0.02461
fAsym	-0.17359
fM3Long	-0.19341

Table 4: Pearson Correlation Coefficients for features of dataset

Cut-off values of 0.01, 0.03, and 0.07 were chosen to test the model's performance upon the discarding of certain lower-correlation attributes from the dataset being trained on. With a cut-off of 0.01, features fM3Trans and fConc1 are discarded. With a cut-off of 0.03, fConc is also removed. With a cut-off of 0.07, fDist is removed, leaving only features fAlpha, fLength, fWidth, fSize, fAsym, and fM3Long left to be trained on. Random Forest and Support Vector Classifier models with optimized parameters chosen earlier were trained on versions of the dataset for each of these cut-off values, and accuracy was calculated at each one.

Principal Component Analysis

Principal Component Analysis (PCA) was the last form of attribute selection performed on the dataset. PCA was done using the Waikato Environment for Knowledge Analysis (WEKA) platform. The PCA decided on seven attributes, or principal components, each of which are linear combinations of the original features of the dataset, that were used as a new set of features for the model to be trained and tested on. A parameter-optimized Random Forest and Support Vector Classifier models were trained and tested on this new set of attributes.

For each of the methods described above, accuracy, model training time (for the 15216 instances), and average predicting time (time to predict all 3804 instances / 3804) were determined.

5. Results and Discussion

Random Forest vs. Support Vector Classifiers

The following are the resulting confusion matrices for the hyperparameter-optimized Random Forest and Support Vector Classifier machine learning models:

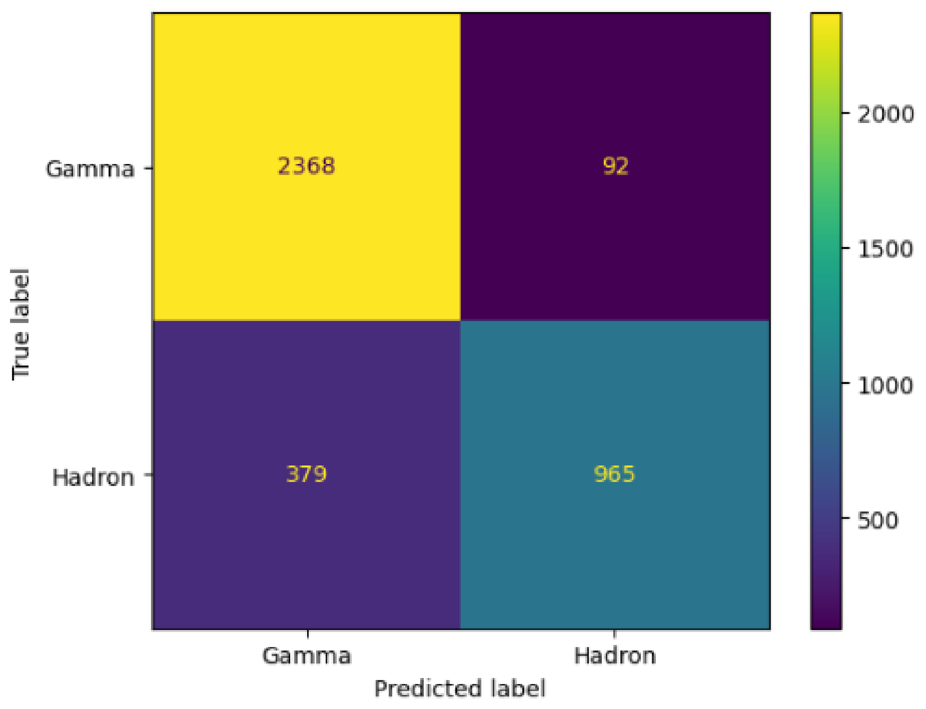

Figure 3: Confusion matrix for parameter-optimized Random Forest Classifier model

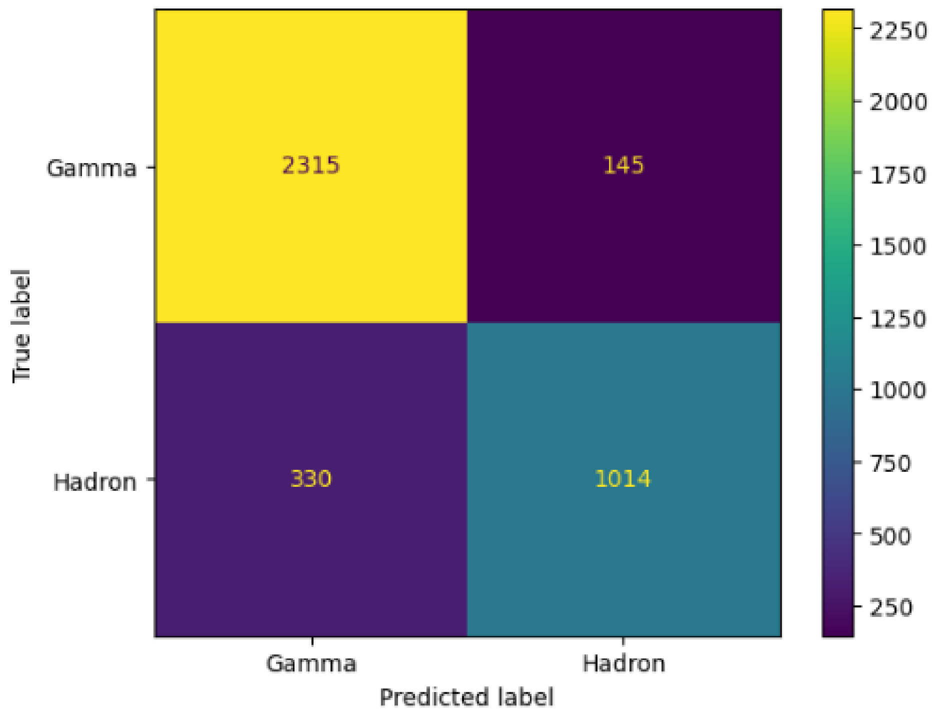

Figure 4: Confusion matrix for parameter-optimized Support Vector Classifier model

As can be seen in the confusion matrices above, while accuracy was similar between the Random Forest and Support Vector Classifier models (88.22% and 87.62%), the Random Forest Classifier model was slightly worse at classifying gamma ray showers and slightly better at classifying hadronic showers, when compared with the Support Vector Classifier model that was slightly better with gamma ray showers and slightly worse at classifying hadronic showers.

Pearson Correlation Coefficient Cut-offs

The following are resulting accuracies when training Random Forests and Support Vector Classifiers on updated versions of the dataset using cut-off values of 0.01, 0.03, and 0.07 to perform attribute selection, plotted on a line chart.

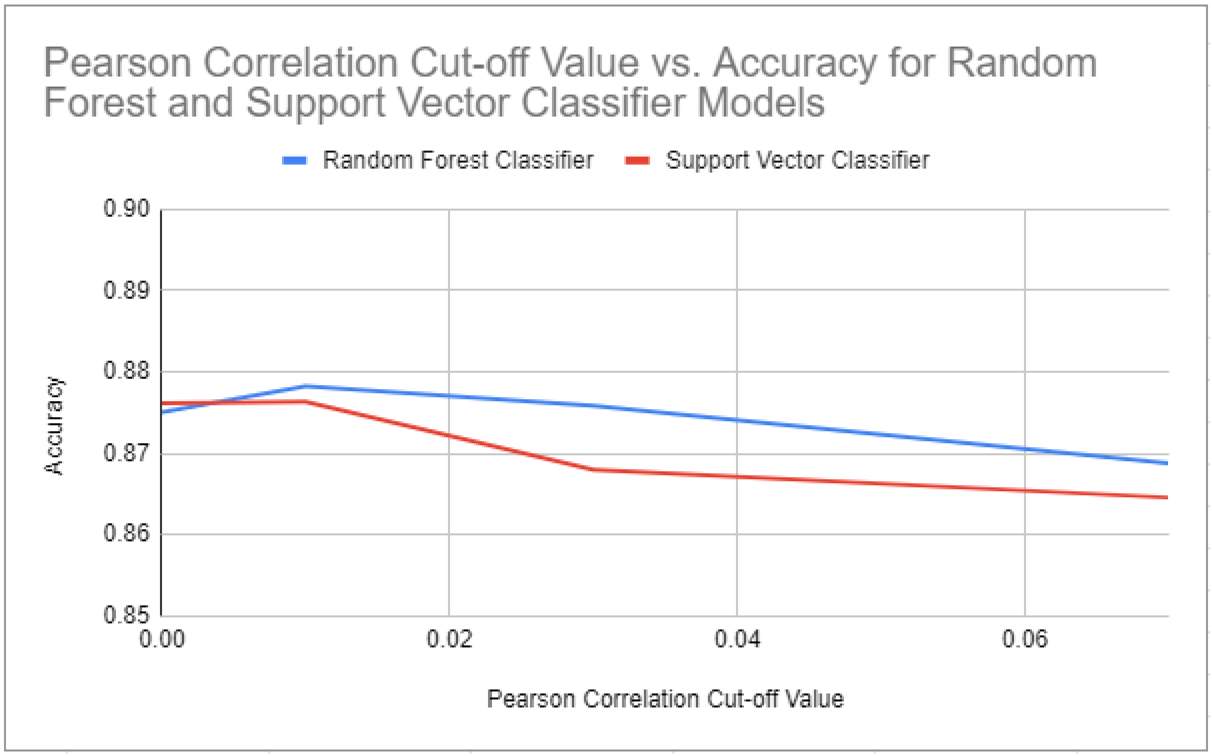

Figure 5: Pearson Correlation Cut-off Value vs. Accuracy for RF and SVC Models

One can see by the trend shown in the graph that the initial cut-off of two features with a cut-off value of 0.01 resulted in an increase in accuracy, while further cut-offs began to diminish model performance. What is surprising about these results is that not only did the first cut-off reduce model complexity without harming performance, but performance was actually increased as a result. If one were to have a perfect dataset (infinite data), using more features will always help, no matter how much or little predicting power each feature has. However, since this dataset is only made up of 19020 instances, it is likely that the reason that the model performed better without the "lower predicting-power" features is that the model did not have enough data to realize on its own that these features were essentially just noise in our data. Since the model did not have enough training to find this out, that is why manually removing the features ourselves was so helpful.

Overall Model Comparisons

		Accuracy	Time to Train (s)	Time to Test (μs)
Models	Unoptimized SVC	87.12%	4.62	525
	Parameter-Optimized SVC	87.62%	6.28	462
	Parameter-Optimized Random Forest Classifier	87.51%	7.30	19.3
	SVC Bagging Ensemble	87.80%	70.59	2670
	SVC and RFC Stacking Ensemble	88.04%	74.31	298
	Pearson Correlation Cut-off = 0.01, SVC	87.64%	5.56	243
	Pearson Correlation Cut-off = 0.01, RFC	87.83%	5.79	16.7
	Pearson Correlation Cut-off = 0.03, SVC	86.80%	5.43	251
	Pearson Correlation Cut-off = 0.03, RFC	87.59%	5.96	17.3
	Pearson Correlation Cut-off = 0.07, SVC	86.46%	5.32	255
	Pearson Correlation Cut-off = 0.07, RFC	86.88%	5.79	19.2
	PCA SVC Model	82.81%	7.60	304
	PCA RFC Model	82.18%	6.08	18

Table 5: Accuracy, training time, and testing time for all models tested

While it also took the longest time to train and test, the ensemble Stacking Classifier yielded the highest accuracy of all the models we tested. The base SVC model took the least amount of time to train, which makes sense because that was the most raw and unchanged model of all the ones used. The most significant result was, however, that we were able to increase the accuracy of both the Random Forest and Support Vector Classifier models by reducing the number of features used for training and testing. This means that model complexity was reduced and model performance was increased at the same time; the goal of our research was accomplished. Not only that, but the Random Forest Classifier trained on data with a Pearson Correlation Coefficient cut-off value of 0.01 also had the fastest prediction time, meaning that in real-life scenarios the model would be extremely good at differentiating between a gamma and hadronic shower given a Cherenkov telescope image, it would do it in under 17 microseconds, and it would not even require all of the features of the image to do so with high accuracy.

6. Conclusions

The goal of our research was to find a way to use a simple sklearn machine learning model to distinguish between gamma and hadronic showers with high accuracy and the least number of attributes possible. Methods including Random Forests, Support Vector Machines, Ensembling, and Dimensionality Reduction were attempted to try and reach this goal. The highest accuracy we attained was 88.04% using a Stacking Classifier Ensemble method that paired a parameter-optimized Random Forest model with a parameter-optimized Support Vector Classifier model. In addition, using a Random Forest model with a Pearson Correlation Coefficient cut-off value of 0.01, we derived a subset of the features that, once trained upon, yielded a higher accuracy than that of the same model being trained on the entire dataset. Specifically, the accuracy of the Random Forest Classifier model went from 87.51% when trained on the entire dataset, to 87.83% when trained on the subset of the dataset not including the two features that were excluded via the cut-off value of 0.01.

Therefore, our goal of maximizing accuracy while reducing dataset dimensionality was successful. This was only possible because, as the dataset only consists of 19020 instances, it is possible that there was not enough data to train on for the model to realize that the features themselves did not have predicting power. Because of that, the manual removal of these features with low predicting power resulted in an increase in model performance. A potential avenue for further research would be the acquisition of more data, potentially artificially. Since there are not many more detected gamma or hadron showers than the ones present in the dataset, it may be useful to search for ways to generate more data some artificial way in order to be able to make better use of all the features, even the ones that do not seem to hold as much predicting power as others.

Acknowledgments

We would like to express our gratitude towards the Inspirit AI program for making this research possible in the first place.

References

[1] Hillas, A.M., 1996, “Differences Between Gamma-Ray and Hadronic Showers,” *Space Science Reviews*, 75, pp. 17-30.

[2] Akhperjanian, A., and Sahakian, V., 2004, “Performance of a 20 m diameter Cherenkov imaging telescope,” *Astroparticle Physics,* 21, pp. 149-161.

[3] Schönfelder, Volker, 2013, *The Universe in Gamma Rays.* https://www.google.com/books/edition/The_Universe_in_Gamma_Rays/z00iCQAAQBAJ?hl=en&gbpv=1&dq=the+universe+in+gamma+rays&pg=PA1&printsec=frontcover

[4] Alfaro, R., et al., 2022, “Gamma/hadron separation with the HAWC observatory,” *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 1039.

[5] Bock, R.K., et al., 2004, “Methods for multidimensional event classification: a case study using images from a Cherenkov gamma-ray telescope,” *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 516, pp. 511-528.

[6] Albert, J., et al., 2008, “Implementation of the Random Forest method for the Imaging Atmospheric Cherenkov Telescope MAGIC,” *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 588, pp. 424-432.

[7] Luo, S., et al., 2020, “An investigation on the factors affecting machine learning classifications in gamma-ray astronomy,” *Monthly Notices of the Royal Astronomical Society*, 492, pp. 5377–5390.

[8] Kecman, V., 2004, “Support Vector Machines Basics,” The University of Auckland, School of Engineering, Auckland, New Zealand.

[9] Boinee, P., Angelis, A., and Foresti, G.L., 2005, “Ensembling Classifiers – An Application to Image Data Classification from Cherenkov Telescope Experiment,” *Proceedings of World Academy of Science, Engineering and Technology*, 7, pp. 394-398.

[10] Yang, Q., 2021, “Linear correlation analysis of ammunition storage environment based on Pearson correlation analysis,” *Journal of Physics: Conference Series*, 1948.