Advanced Machine Learning
Block 2

Maria Paraskeva
Omar Rúbio López

UPC - FIB
November 2023

1 Introduction

The Major Atmospheric Gamma-ray Imaging Cherenkov (MAGIC) telescopes are two large telescopes, each with a 17-meter diameter. They work together to detect cosmic gamma rays, which are a type of high-energy radiation from space. These telescopes have been operational since 2003, and they began working in tandem in 2009.

When cosmic gamma rays or other charged particles from space hit the Earth's atmosphere, they create a kind of light show known as Cherenkov light. This light is very brief and travels towards the ground, creating a specific pattern that the MAGIC telescopes can detect and capture with their cameras.

The telescopes have special cameras that can quickly pick up these light patterns. When both telescopes detect the same pattern at the same time, they collect and save this data. Using various algorithms, scientists can then figure out where each gamma ray came from, the time when it arrived, and how much energy it had. This process involves removing background noise (like light from the night sky), analyzing the light pattern, and using sophisticated techniques to distinguish between gamma rays and other particles, like cosmic rays.

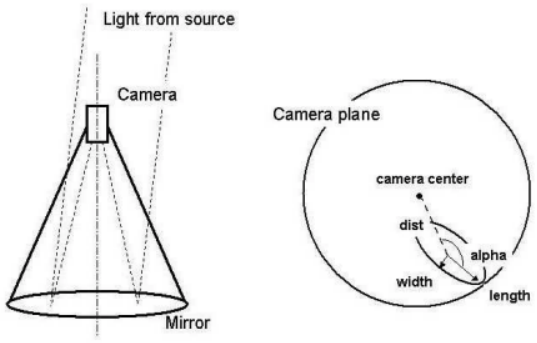

Figure 1: Geometric feature extraction from gamma event

The data we are using for this project is a simulate registration of high energy gamma particles in a ground-based atmospheric Cherenkov gamma telescope using the imaging technique, by a Monte Carlo program in the work by Heck et al. [4], containing the following variables:

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]
11. **class:** g,h - gamma (signal), hadron (background)

The task in this dataset is to accurately predict if the telescope is receiving an actual signal or it is just some random noise.

1.1 Previous Work

In Bock et al. [1] some work was done with ML methods but under a slightly different context. In this paper, they used the Cuteval method [2] that involves numerical maximization resulting in the highest possible value of Q, a quality parameter. For instance, $Q = \frac{S}{\sqrt{B}}$, where S is the number of signal events, and B is the number of background events after selection.

In this work, several methods were tried with this approach, like Random Forest, KNN, SVM, etc., but the paper admits that the work on SVM was poor and only a rudimentary first attempt was made.

The best results were obtained using a Random Forest, performed by using bagging and random split selection. Bagging involves sampling with replacement from the original training dataset, and random split selection occurs during tree growth. The root node was split based on a randomly selected subset of image parameters, using the parameter with the smallest Gini-index. The 50 resulting unpruned trees were combined using a simple arithmetic mean, which proved effective with a sufficiently large number of trees. Attempts with weighted combinations showed improved results only for a very small number of trees (less than 10), aligning with the understanding that a large forest is less sensitive to variance than bias.

The route we followed when performing Random Forest was different, as we believed it would yield better results. They can be found in Subsection 3.2.

We could say that our study is more focused on kernels and tree methods, leaving out Neural Networks and other more general classification methods used in the paper as they are out of the scope of our subject.

In this work we proposed possible dimensionality reduction techniques, model explainability, better hyperparameter selection and some insights in the data without being experts in the matter.

2 Data Exploration

In order to get a better view of our data, we first need to acquire a comprehensive overview of the data's structure and basic statistics. This can be achieved through a variety of visualization techniques, such as plots and correlation matrices.

Before proceeding to these techniques, we might make some speculations about the expected correlations. We would expect the variables fLength and fWidth to be positively correlated since they are both describing the shape of the ellipse. These two might also be associated with fSize as larger objects and thus, larger fSize, could mean longer major axes as well as wider minor axes. Another pair of variables that is expected to show correlation is fConc and fConc1 as they both involve ratios of pixel values to the size of the measured object, so a positive relationship should be expected.

The variables fAsym and fM3Long are candidates for correlation as well. The first measures the distance from the highest pixel to the center, projected onto the major axis. If an object is asymmetric, meaning the distribution of the pixels is not balanced around the major axis, this could result in a higher fAsym value. The second variable, on the other hand, involves the third root of the third moment along the major axis. The moment represents the distribution of pixel values, and the third root introduces some degree of normalization. If there is asymmetry along the major axis, it might be reflected in both a larger fAsym and a larger fM3Long, so a positive correlation is expected.

Finally, fM3Long and fM3Trans might show some correlation as well. These features both involve the third root of the third moment but along different axes. They both capture aspects of the shape and skewness of the distribution, so if there is a consistent pattern of skewness in the distribution of pixel values, it might be shown in both variables.

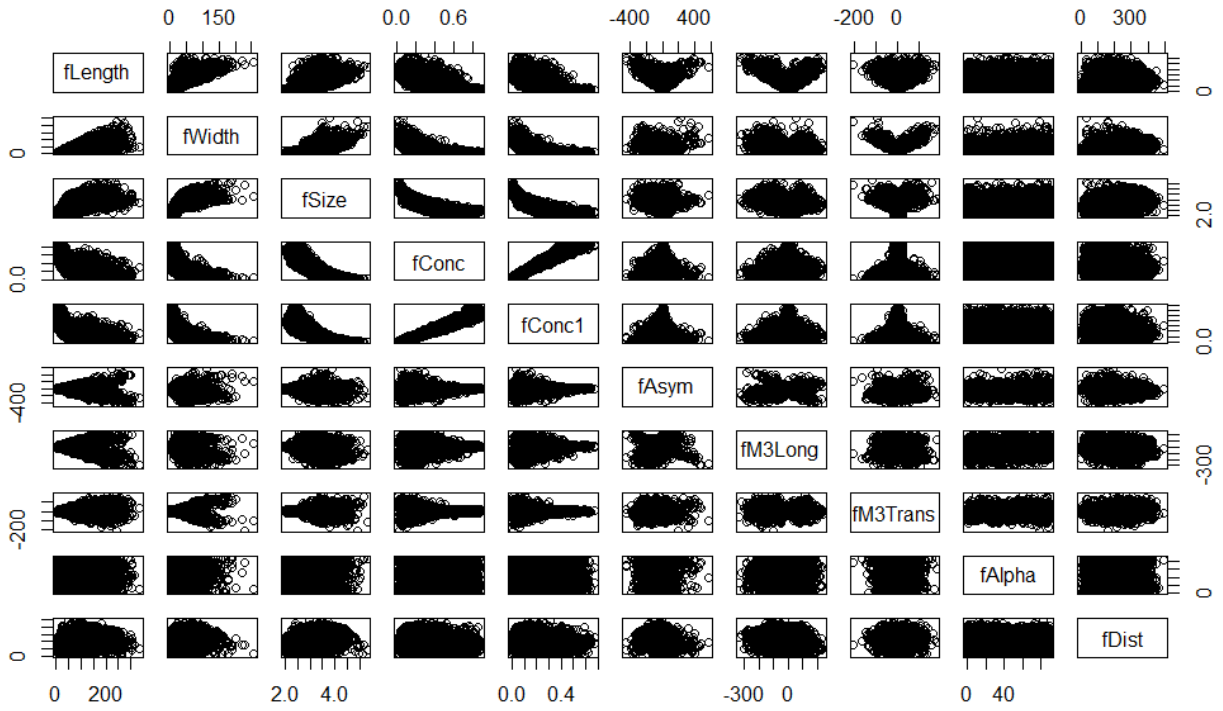

Figure 2: Pairplot of the predictors

As a first step we created scatterplot matrices of the predictors which lie in the first ten columns of the dataset. The pair plot in Figure 2 visually explores the pairwise relationships between these predictors, which will allow us to make some speculations about possible relationships. The diagonal of the pair plot displays the name of each predictor and each point on the scatterplots correspond to an observation in the dataset.

A first observation shows that certain pairs of predictors exhibit a clear linear relationship, forming a distinguishable straight line pattern. The variables fConc and fConc1 are positively correlated as their pair plot show a positive slope. An increase in fConc will result in an increase in fConc1 as well.

The variables fLength and fWidth could also suggest some correlation even though the plots are not that clear, maybe using some log transformations.

Some relationships like fConc1 and fM3Trans/fAsym have a high variance at low fConc1 values to later decrease. And we have the inverse relationship with fLength and fM3Trans, with higher fLength we have higher variance.

Other relationships like fSize with fConc or fWidth with fConc seem to have a type of reciprocal squared.

Further visualization techniques should be implemented lest we make any incorrect deductions and, as the rest of the pair plots are difficult to decipher, we proceed to the creation of a heatmap shown in Figure 3.

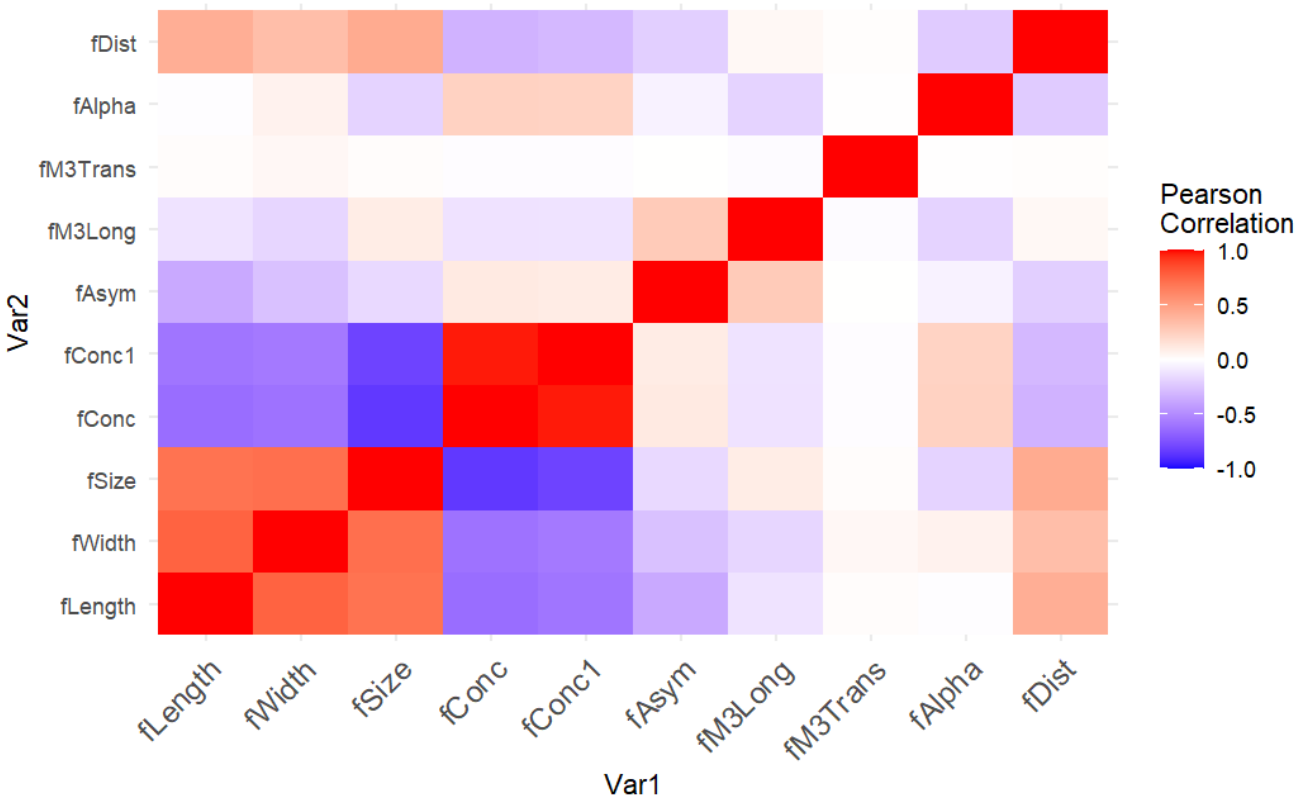

Figure 3: Heatmap of the Pearson Correlation of the predictors

The above heatmap was created using the Pearson correlation coefficients between all pairs of the ten predictors. Each cell in the heatmap represents the correlation of the two selected variables. The color of each cell indicates the strength and direction of the correlation, with a gradient from blue showing a negative correlation to red showing a positive correlation.

As expected, the highest correlation turned out to be between fConc1 and fConc, with their cells having almost the same hue of red indicating a correlation very close to 1.

The group of variables including fLength, fWidth, and fSize also show a high positive correlation among them suggesting a number around 0.5 on the Pearson scale and proving our expectations correct. They are negatively correlated with the fConc1 and fConc1 pair, as the hue on the cells indicate a -0.5 correlation, with the fSize cell having a darker hue of blue. The first three variables also show a connection to fDist and it would make sense as this variable measures the distance from the origin to the center of the eclipse. Generally, as the major or minor axis of the ellipse (represented by fLength or fWidth) increases, the distance from the origin to the center of the ellipse (fDist) might also increase. Regarding the variable fSize, larger ellipses (larger fSize) might generally have a greater distance from the origin to the center (fDist), hence the correlation in the heapmap appears positive.

The one variable that shows zero correlation with the rest is fM3Trans, showing no statistical independence among the other variables. This result is surprising, as a positive correlation was expected between this and the variable fM3Long.

Another unexpected correlation is between fAlpha and the pair of fConc and fConc1. The correlation does not appear very strong but it could be explained from the fact that as the major axis rotates (determined by fAlpha), it may be influencing the distribution of pixel values along the major axis, impacting both fConc and fConc1. In addition, changes in the size of the object (related to the major axis and fSize) might influence how pixel values are concentrated, leading to a positive correlation with the ratios fConc and fConc1.

In order to gain further insights, we proceeded to the implementation of PCA (Principal Components Analysis) and we plotted the first two dimensions as can be seen in Figure 4

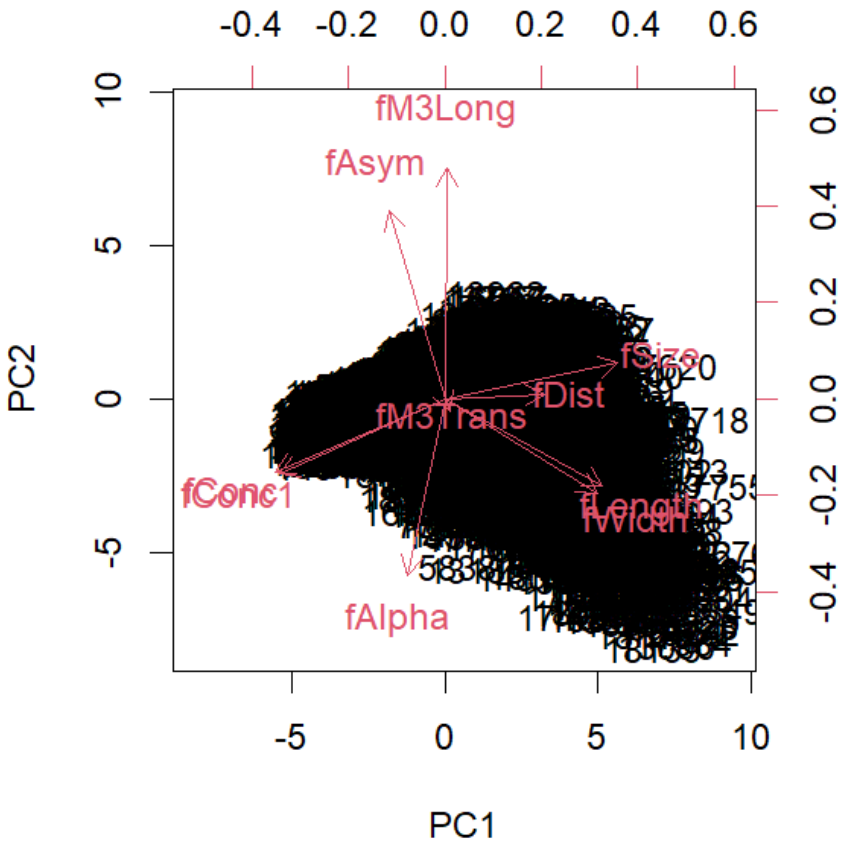

Figure 4: First two dimensions of the PCA

PCA aims to reduce the dimensionality of the data by transforming it into a new set of uncorrelated variables called principal components. We plot the first two dimensions so the data are visualized in a new coordinate system defined by the first two principal components.

This might be useful in a context where we are not able to compute the Kernel Matrices due to high cost, and be used as a dimensionality reduction technique. We will use it as an alternative if the computational cost comes out as too expensive in the following sections.

The first principal component (PC1) represents the direction of maximum variance in the data and is represented by the x-axis of the plot. The second principal component (PC2) is orthogonal to PC1 (y-axis) and represents the second highest variance. The proportion of total variance explained by each component is typically provided in the PCA results. By plotting the data in this reduced-dimensional space, we are able to see how much of the total variance is captured by the first two components. This is usually done for plotting, as we can represent data in a 2-dimensional space, however, we can add as many dimensions as variables.

We will focus on the angles between vectors in the PCA plot which reflect the correlation between the corresponding variables. Variables that are positively correlated will have similar angles in the plot, while negatively correlated variables will have opposite angles.

Starting from the variable fM3Long, we can see its vector being closer to that of the fAsym variable, forming an angle of about 30°. The light blue hue of the cell of variable fAlpha against the two aforementioned variables in Figure 3 is now better visualized as the vector of fAlpha is almost opposite of the vectors of fM3Long and fAsym. The variable fAlpha itself does not appear closely correlated to any variable, although it is situated in the same quadrant on the x-y axis with the fConc-fConc1 pair, indicating a low positive correlation in their directional alignment.

Moving clockwise, the correlation between fSize and fDist is more visible in this plot, with their vectors creating an angle of about 10°. This correlation was higher than expected, as the heatmap above was showing a red hue

closer to 0.5. fConc and fConc1 are almost forming one vector, a result that was highly expected, and their vectors are opposite fSize and fDist, explaining the blue hue in the respective heatmap cells.

Considering how correlated fSize, fDist, fLength, and fWidth were in Figure 3, it is interesting to see that they actually form a 45° angle in pairs of two; fSize closely correlated to fDist against fLength which is highly correlated to fWidth. This observation gives us new insights of the behavior of these predictors, allowing us to make decisions regarding the preprocessing of the data set.

Lastly, we can hardly see the vector of fM3Trans, even though until now our knowledge from the previous plots shows that it is an independent variable.

The first component seems to be differentiating between objects or phenomena that are large and extended (high fSize, fDist, Length, and Width) and those that are less concentrated (low fConc). This could represent the difference between large, diffuse objects (like nebulae following astrophysics argo) and smaller, more concentrated ones (like stars).

The second component might correspond to objects or observations that have a greater distance from the highest pixel to the center (fAsym) and a more pronounced third moment along the major axis (fM3Long), while also having a smaller angle with the vector to the origin (fAlpha). In practical terms, this could represent a specific physical or geometrical characteristic that might relate to the shape and orientation of celestial objects.

After plotting the first two principal components, we decided to plot the cumulative variance plot which can be found in Figure 5.

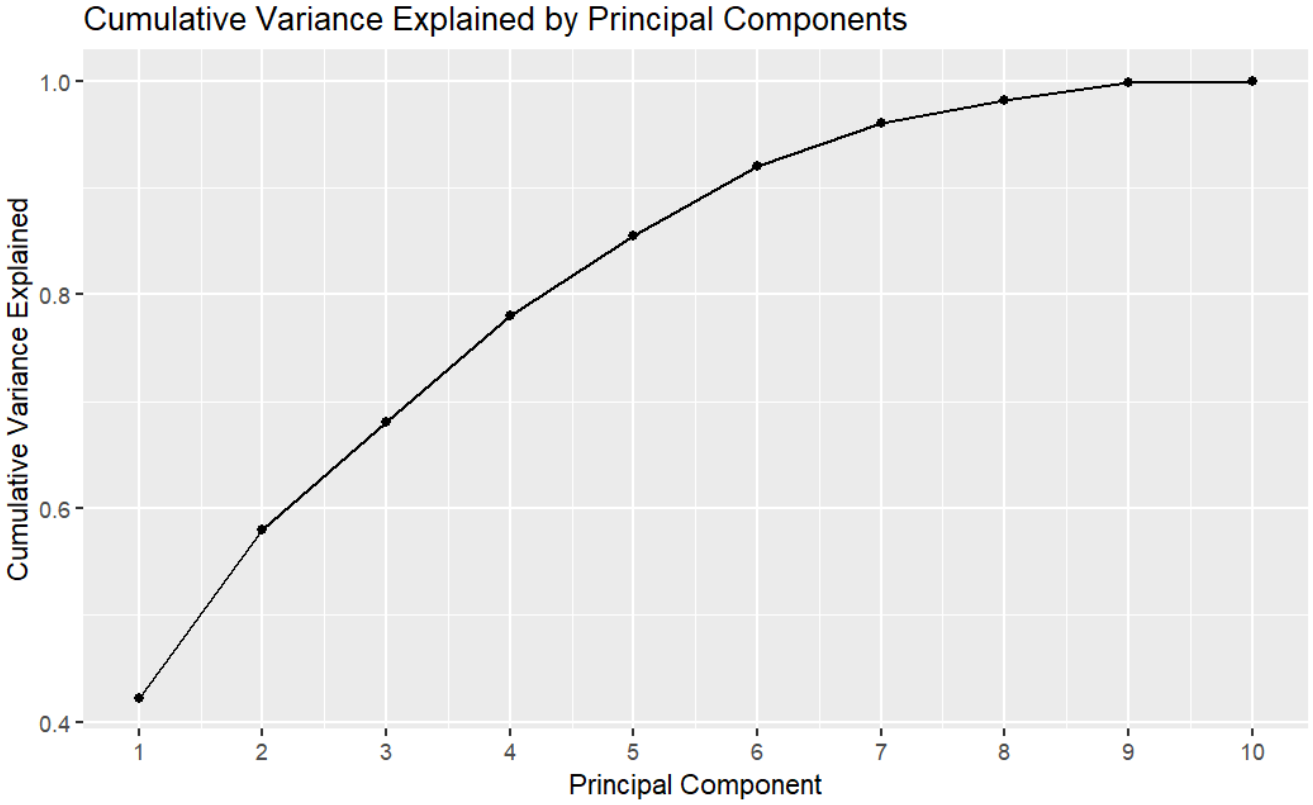

Figure 5: Cumulative variance for the components of the PCA

In order to get information about how much of the total variance in the dataset is explained by each principal component, we observe the results of the cumulative variance for the components of the PCA plot above. The cumulative variance is the sum of the variances explained by all the principal components up to a certain point.

The x-axis of the plot represents the number of principal components which, in our case, goes up to 10 as we have ten predictors, while the y-axis shows the cumulative variance. The value of the cumulative variance ranges from 0% up to 100%, with the highest value indicating that the included principal components collectively explain the entire variance in the dataset, fully reconstructing the original dataset.

Our plot shows that the first two components explain 60% of the whole dataset and by adding two more we reach almost 80%. After the sixth component the curve start to flatten, which suggests that additional principal components will contribute less to the cumulative variance.
This can be used for further processing if the algorithms get stuck on the calculations.

Focusing more on each individual predictor, we created their boxplots since they are all numeric values. The results can be seen in Figure 6.

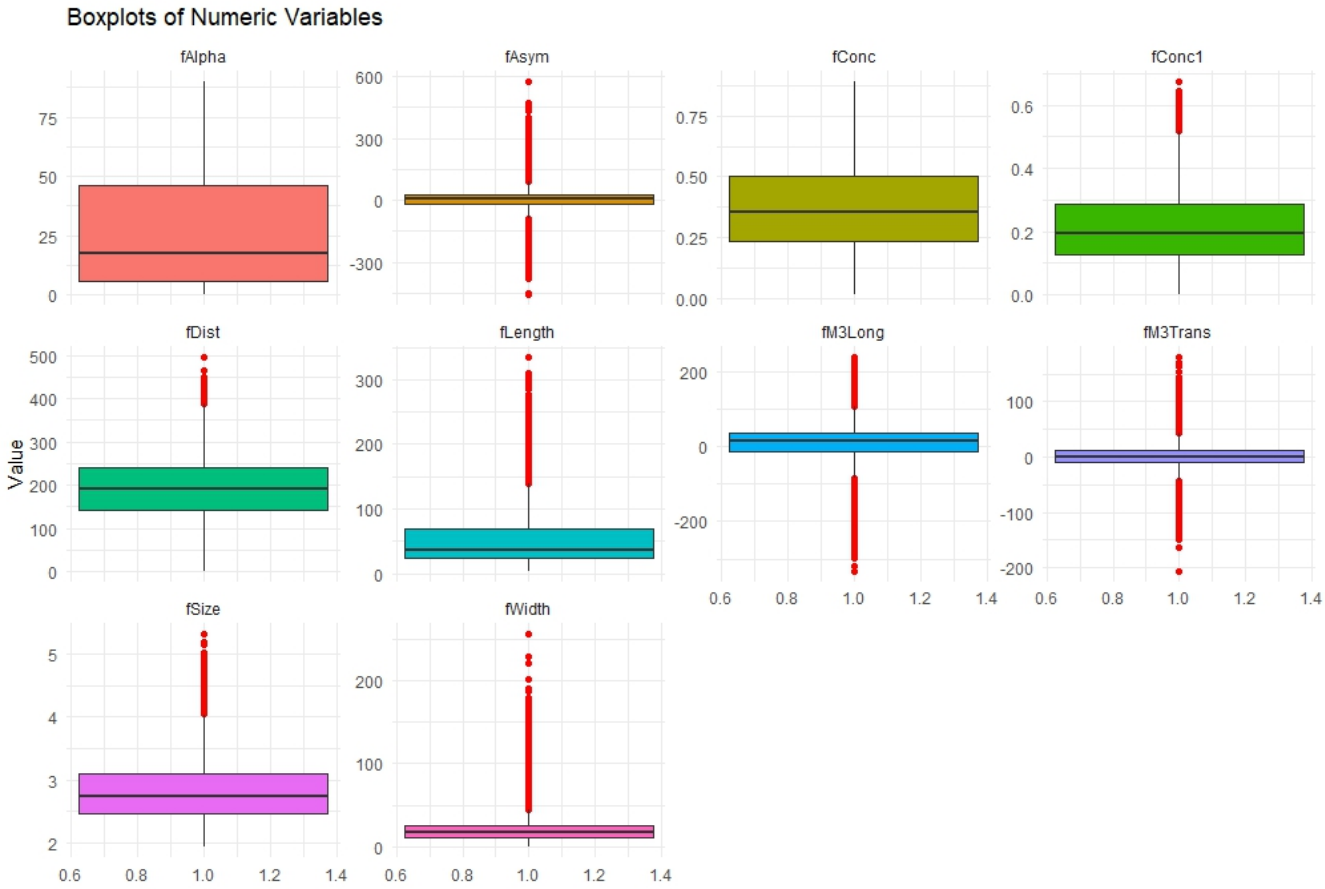

Figure 6: Boxplot of the predictors

The boxplots above will provide a visual summary of each predictor's distribution. By looking at each box we can figure out the interquartile range (IQR), which is the range between the first and the third quartile (Q1 and Q2 respectively). The heights of the boxes indicate the spread of the middle 50% of the data. The line inside each box represents the median (Q2), which is the middle value of the dataset. The whiskers extend from the edges of the box to the minimum and maximum values within a certain range. The exact definition of the range can vary as we will see below. Finally, the individual points beyond the whiskers are considered outliers as they are data points that fall significantly outside the typical range.

Starting by taking a look at the smaller boxplots of our dataset, we notice that the variables fAsym, fM3Long, fM3Trans and fWidth share some similar features. All of their medians appear close to zero apart from that of fWidth, which appears closer to a value of 20 and fM3Long which appears close to 10. This shows us that, on average, the distance from the highest pixel to the center (fAsym) as well as the third root of the third moment along the minor axis (fM3Trans) are not significantly biased towards one direction. Also, the minor axis of the ellipse is around 20 mm while the third root of the third moment along the minor axis is around 10 mm on average. The means of fM3Trans and fWidth are in the center of the boxes which suggests symmetry, while that of fAsym and fM3Long lean towards the higher end of their range, a fact that indicates skewness. The presence of outliers is apparent in both sides of the boxes for all variables, except for fWidth which shows extreme values only on the positive side of the distribution, indicating values that are significantly higher than the majority of the data and representing cases with exceptionally larger widths.

The objects that the telescope is observing appear to be quite large, as both fLength and fWidth have outliers on the upper part of their boxes. The median of fLength is around 40, meaning that, on average, the major axis of the ellipse is around 40 mm. The length of the box shows greater spread than fWidth as it is longer, showing that the middle 50% of the widths are concentrated in a narrower range than the middle 50% of the lengths. This also shows greater variability in the major axis lengths compared to the minor axis widths. Finally, the median of the lengths leans towards the lower end of the box indicating skewed data as well as that the median is less influenced by extreme values compared to the mean.

Moving on to variable fDist, the average distance from the origin to the center of the ellipse is around 200 mm and the data distribution appears symmetric as the median is near the center of the box. There are some outliers on the upper part of the box starting from 370 mm with one single observation measuring 500 mm representing a notable deviation from the typical observed distances even though the majority of the observations are clustered around the mean. It is implied that these specific Cherenkov light patterns are detected at spatial locations significantly farther from the origin than the majority of observations.

The variable fSize almost shows symmetric distribution as well, as the median is ever slightly leaning toward the lower end of the box, with a value close to 2.75. The interquartile range spans from around 2.4 up to 3.2, where the majority of the values are concentrated. There are some outliers with values from 4 to 6 meaning that the logarithmic measure of the sum of pixel content in the ellipse is much higher in these observations than the middle 50% of the other logarithmic measures.

Checking the fAlpha's boxplot, we can get an idea about the orientation of the Cherenkov light pattern. Its median is around 15 showing that, on average, the angle of the major axis with the vector to the origin is about 15°. The median is closer to the lower end of the box showing skewness and that there are some observations with an angle less than the median, indicating a tendency for the major axis to align more frequently with the vector to the origin. The box span shows us that the interquartile range is around 40 degrees with its values starting from 10 up to 50. There are no visible outliers although the top whisker of the boxplot is quite long. This suggests that there is a notable range of values in the upper part of the dataset, but the data points in that range are not considered outliers based on the typical criteria for identifying outliers.

Another variable whose boxplot indicates the same observation regarding outliers is fConc. This variable measures ratios related to the concentration of pixel values in the Cherenkov light pattern and, in particular, the ratio of the sum of the two highest pixel values to the log sum of the content of all pixels in the ellipse. This ratio takes small values with its a median of around 0.7 and a interquartile range from 0.25 to 0.50. We could say that the distribution of the data is symmetric as the median stays in the middle of its box. Compared to its highly correlated counterpart, fConc1, we observe a higher median and a larger IQR, as the latter's median lies at only 0.2 while its values span from 0.12 to around 0.3, showing a shorter box with a slightly skewed median to the lower part of the box.

On the other hand, as fConc1 measures the ratio of only the highest pixel value within the Cherenkov light pattern relative to the overall content of the ellipse, its boxplot shows some distinguishable outliers. Specifically, the range of the outliers is from 0.5 up to almost 0.7, values considered extremely high compared to the IQR. These observations indicate Cherenkov light patterns where a single pixel has an unusually high intensity compared to the overall content of the ellipse and, as they could possibly represent rare or significant astronomical events where the Cherekov light is exceptionally concentrated in a single pixel. As these outliers might affect our results, fConc1 is a good candidate for removal as it is also significantly correlated to the variable fConc.

As a final step in our data exploration, we present density plots for each predictor variable, offering a visual examination of their distributions and potential discriminative patterns between gamma (signal) and hadron (background) events. The plots can be found in Figure 7 below.

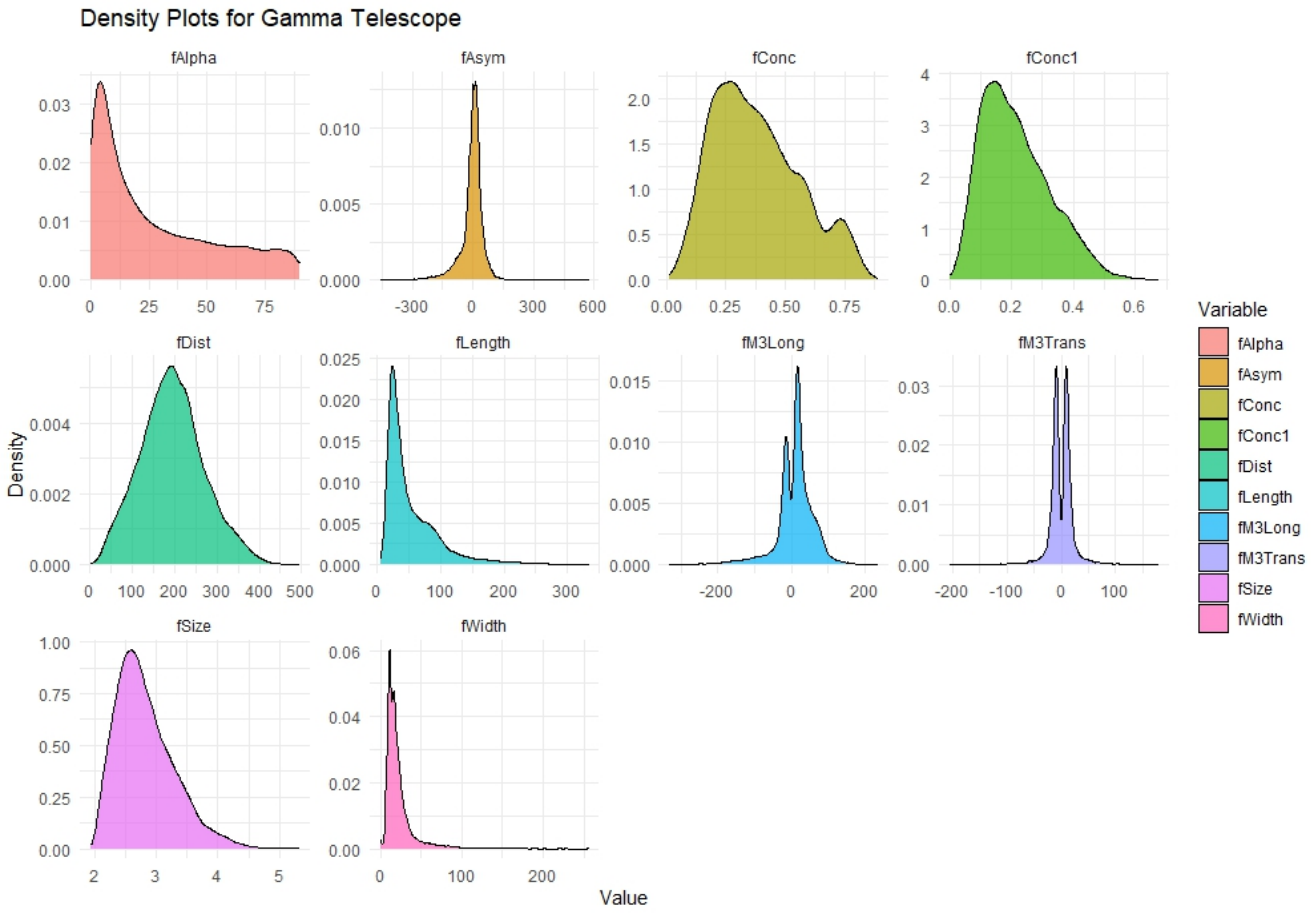

Figure 7: Density Plots for the predictors

Density plots, also known as kernel density plots, are graphical representations of the distribution of a continuous variable. They provide a smoothed estimate of the probability density function of the data, allowing us to visualize the underlying shape of the distribution. In a density plot, a kernel function is placed at each data point, and these individual kernels are then summed or averaged to create a smooth curve. The resulting plot gives us a sense of the data's distribution, showing where values are concentrated and how the density changes across the range of the variable. In our case and in order to capture the behaviour of all the variables, we generated a grid of density plots with each variable represented in a separate facet.

Starting with the highly correlated pair of variables fConc and fCon1, we notice a small positive skewness in the plot of the former that was not clearly visible in the boxplot of Figure 6. The latter's positive skewness shown in its boxplot is now confirmed, seeing that the right tail is longer than the left. We can also notice how fConc's wider curve indicates higher variability than fConc1, a fact also shown by the height of their respective boxes in Figure 6.

Continuing with the density plots of the also correlated group of variables fDist, fLength, fSize, and fWidth, we first notice how apparent the outliers of fLength and fWidth are. Both appear to have a longer right tail, which indicates small quantity of data points with values higher than usual. The concentration of the data in both plots are around its median and their short boxes are now explained by the slender shape of their curves. The tails also show that both distributions are right-skewed, a fact that was not noticeable from the boxplot of the variable fWidth. Regarding their curves, fLength has slightly higher variability as its curve is a bit wider than than fWidth's. The variables fDist and fSize, though, appear to have even higher variability than the first two, as their curves are evidently wider, an observation also supported by their respective boxplots (longer boxes). FSize has a positive skewness that was not visible in its boxplot while fDist has only slightly more values concentrated on the left part of the plot.

The density plot of the variable fAlpha also confirms our previous observations. With a positive skewness and the data concentrated to the left side of the plot, the tendency for the major axis to align more frequently with the vector to the origin is now evident. The right-skewed distribution suggests that a majority of the data points have lower values, as indicated by the longer and fatter tail extending towards the higher values. Interestingly, the

right side of the distribution shows a distinctive shape similar to a logarithmic plot, characterized by a swift initial descent which confirms that a significant proportion of the data has lower fAlpha values.

Finally, the last three predictors, fAsym, fM3Long, and fM3Trans show similarities in their density plots. All three plots have their values concentrated within a narrow range, which confirms their small spread and short boxes in Figure 6. The outliers are also visible when looking at their left and right tails, similar to the red dots on their boxplot's whiskers. The variable fAsym has a longer right tail which shows that its distribution is right-skewed, a fact shown by its boxplot as well. A new observation here would be a possible bimodal distribution in the variables fM3Long and fM3Trans. They both have a dip around zero, indicating that the lower middle values are taken around zero and there are two prominent peaks in the data. Each peak corresponds to a mode which is a high density region. Both of these variables represent the third root of the third moment, one along the major axis and the other along the minor axis of the ellipse. So, the distribution of pixel content appears to be divided in different subgroups.

For the evaluation of our models, we decided to split our dataset in an 80-20 ratio, ending up with the following number of observations in each split:

	g	h
Train	9865	5350
Test	2467	1338

Table 1: Target Variable Count for Train and Test Data

Note that this is a generated dataset and the number of h events is underestimated. In the real data, the h class represents the majority of the events. 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. For this reason, our evaluation will be comprised by the comparison of different ROC curves, as it will be seen later in our analysis.

3 Modelization

3.1 Preprocessing

In the preprocessing phase of our analysis, we identified strong correlations between certain features: fConc1 is highly correlated with fConc, and fLength shows a similar relationship with fWidth. When features are highly correlated, the regularization might distribute weights among these features, making it harder to interpret the importance of individual features. Also, when using a kernel, the relationships between features can become even more intertwined, and correlated features in the original space might behave unpredictably in the kernel-induced space. For these reasons, we chose to remove these variables.

For splitting our dataset, we adopted an 80-20 shuffled split, stratified based on the target variable to ensure a balanced representation of classes in both subsets, as we have seen previously that the dataset is unbalanced. The 80-20 split is widely adopted empirically, but a possible explanation of this can be found in Gholamy et al.[3] and some basic reasoning of why we need this split.

We used the tidymodels library for our preprocessing tasks. This library has proven effective and offers a streamlined approach to preprocessing. It allows us to save the entire preprocessing pipeline, which includes various steps, and apply it to new, unseen data easily.

Our preprocessing pipeline primarily consisted of two steps: centering and scaling. By centering, we adjusted the data so that each feature has a mean of zero. Scaling ensured that each feature has a variance of one. This standardization is crucial for models that are sensitive to the scale of the input features and usually converge faster.

Additionally, we implemented a step to remove multicollinearity by excluding features with a correlation higher than 0.9. Although we had manually removed some features earlier based on their correlation, this step acted as a failsafe to ensure no highly correlated features were included in the model.

In Table 2 we can see the outlier ratios for each numeric predictor. The visualization of the boxplots in Figure 6 helped to successfully identify them.

As we do not have expert knowledge in this domain, we removed these outliers in order to achieve a better average prediction, but further investigation may be needed.

Variable	Outlier Ratio
fLength	0.051051525
fWidth	0.081282860
fSize	0.019295478
fConc	0.000000000
fConc1	0.007886435
fAsym	0.093638275
fM3Long	0.064037855
fM3Trans	0.044689800
fAlpha	0.000000000
fDist	0.006729758

Table 2: Outlier Ratios for predictors (Number of outliers calculated using 1.5 IQR method, divided by total observations)

3.2 Evaluation Methodology

For this project we used the following methods:

- SVM with RBF Kernel (1)
- SVM with Polynomial Kernel (2)
- SVM with Histogram Kernel (3) [1]
- Random Forest with a fixed number of trees (1000, due to time limitations)
- KNN

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2\right) \tag{1}$$

$$K(\mathbf{x}, \mathbf{x}') = \left(\alpha \langle \mathbf{x}, \mathbf{x}' \rangle + c\right)^d \tag{2}$$

$$K(\mathbf{x}, \mathbf{x}') = \sum_i \min(\mathbf{x}_i, \mathbf{x}'_i) \tag{3}$$

Method	Parameter	Values
SVM with RBF	cost	$2^{-5}, 2^{-3}, 2^{-1}, 2^0, 2^1, 2^3, 2^5$
	gamma	$2^{-5}, 2^{-3}, 2^{-1}, 2^0, 2^1, 2^3, 2^5$
SVM with Polynomial	cost	$2^{-3}, 2^0, 2^3$
	degree	2, 3
SVM with Histogram	cost	$2^{-3}, 2^0, 2^3$
Random Forest	mtry	2, 3, 4, 5, 6, 7
	splitrule	gini, extratrees
	min.node.size	1, 5, 10
KNN	k	1, 3, 5, 7, 9, 11, 13, 15

Table 3: Hyperparameter grid for the different Models

For the hyperparameter selection, we used different grids of parameters and applied a 10-fold cross validation (CV). In this case, the metric we used in the code was the accuracy by default, which may seem awkward as we have an unbalanced problem by a ration of 2 to 1, however, we checked the tune history of each model to ensure that no model was overfitting to the majority class. We wanted also to use some custom metrics to ensure this, but as we had some time/computational problems calculating some kernel matrices we had to leave it and only use the accuracy for this part.

Once we selected the hyperparameters for each model, we used several metrics to select the best model:

[1] This is a valid kernel in $\mathbb{R}$, a proof can be found here

- ROC
- Confusion Matrix
- Precision-Recall AUC

Out of these metrics, the most important for us is detecting true positive cases in a balanced way. In research areas like high-energy astrophysics, the primary goal is often to ensure the purity of the signal rather than the completeness of signal detection. Precision aligns well with this goal, as it measures the purity of the identified gamma particles in the dataset.

Finally, remark that our positive class is g. That is, a signal and not background noise.

3.3 Results

Model	Best Hyperparameters	Accuracy
SVM with RBF	$C = 32, \gamma = 0.125$	0.8682892
SVM with Polynomial	$\text{cost} = 1, \gamma = 8, \text{degree} = 2$	0.8216248
SVM with Histogram	$C = 8$	0.547022
Random Forest	$\text{mtry} = 2, \text{splitrule} = \text{gini}, \text{min.node.size} = 1, \text{trees} = 1000$	0.8727574
KNN	$k = 5$	0.8393044

Table 4: Best Hyperparameters results for 10-fold CV on the train set

In the Best Hyperparameters Table (4) the results of our computationally expensive grid search of 10-fold CV can be seen. Here, the Random forest is the clear winner with an accuracy of 0.87, followed by the RBF Kernel with 0.86. We thought the Histogram Kernel would perform better as we performed some experiments with it before, but the data was far from being similar to this, which probably lead to this result of 0.547.

The following results are based on the train set without the use of CV, as we lacked the computational power to re-run all the experiments within our time frame.

In Table 5 we can check the confusion matrix of the RBF Kernel, and in Table 6 some results extracted from it. As we said previously, we aim to have the best precision possible, and in Figure 14 we can see an AUC of 0.867, which is not a bad result at all.

Prediction	g	h
g	9521	1406
h	344	3944

Table 5: Confusion Matrix for SVM with RBF Kernel

Statistic	Value
Accuracy	0.885
95% CI	(0.8798, 0.89)
Kappa	0.7357
Sensitivity	0.9651
Specificity	0.7372

Table 6: Statistics for SVM with RBF Kernel in the train set

The results for polynomial kernel can be found in Tables 7 and 8, with an AUC of 0.8 in Figure 13

Prediction	g	h
g	9432	2249
h	433	3101

Table 7: Confusion Matrix for SVM with Polynomial Kernel

Statistic	Value
Accuracy	0.8237
95% CI	(0.8176, 0.8298)
Kappa	0.5809
Sensitivity	0.9561
Specificity	0.5796

Table 8: Statistics for SVM with Polynomial Kernel

Similarly, the confusion matrix for the Histogram Kernel can be found in Tables 9 and 10 with an AUC of 0.6 in Figure 11

Prediction	g	h
g	24	415
h	9841	4935

Table 9: Confusion Matrix for SVM with Histogram Kernel

Statistic	Value
Accuracy	0.3259
95% CI	(0.3185, 0.3334)
Kappa	-0.0535
Sensitivity	0.002433
Specificity	0.922430

Table 10: Statistics for SVM with Histogram Kernel

These results are pointing to RBF kernel as the winner of the kernel methods, having a better statistic in all the categories, which is no surprise as it is the standard kernel in many ML applications.

Prediction	g	h
g	9480	1370
h	385	3980

Table 11: Confusion Matrix for KNN

Statistic	Value
Accuracy	0.8847
95% CI	(0.8795, 0.8897)
Kappa	0.7359
Sensitivity	0.9610
Specificity	0.7439

Table 12: Statistics for KNN

Our baseline model, the KNN, performed surprisingly well (Tables 11, 12 and Figure 12), similar to the RBF Kernel. After comparing them, we extracted the following conclusions:

As our priority is correctly identifying as many positives as possible, the SVM model may have a slight edge due to its higher sensitivity. Considering both sensitivity and specificity, and the very similar AUC and Kappa values, both models perform almost equally well. The choice may come down to factors like computational efficiency, so in this case, comparing both methods we would choose the KNN.

The Random Forest outperforms every single classifier in the train set with a perfect classification (Table 13, 14 and Figure 15).

A final comparison of the models can be found in Figure 10 with the ROC-AUC, where Histogram is the worse classifier, followed by the SVM with Polynomial Kernel, and KNN tied with the RBF SVM.

Prediction	g	h
g	9865	0
h	0	5350

Table 13: Confusion Matrix for Random Forest

Statistic	Value
Accuracy	1.0000
95% CI	(0.9998, 1)
Kappa	1.0000
Sensitivity	1.0000
Specificity	1.0000

Table 14: Statistics for Random Forest

Model	AUC (Integral)	AUC (Davis & Goadrich)
SVM with RBF	0.8670132	0.8670132
SVM with Polynomial	0.8036667	0.8036667
Random Forest	1	1
KNN	0.8688414	0.8688414
SVM with Histogram	0.6002835	0.6002835

Table 15: Precision-Recall AUC for Various Models

3.4 Test Evaluation

Given the previous results, we will focus on the CV results and the train results. If we check Table 4, Random Forest outperforms all the other candidates with an accuracy of 0.8728, and the RBF Kernel follows with 0.86829.

To further check these results, an evaluation on the train set (Tables 13 and 5) gives a clear winner where the Random Forest perfectly classifies the train set. This could be a symptom of overfitting, but when checking the CV results, it is evident that it outperformed the other methods as well with some more realistic results. So, it will probably be even with the RBF SVM, and, in this context, the CV and the confusion matrix results are enough to select this model.

Statistic	Value
95% CI	(0.8564, 0.8782)
No Information Rate	0.6484
Kappa	0.7004
Mcnemar's Test P-Value	3.165e-15
Sensitivity	0.9339
Specificity	0.7451
Pos Pred Value	0.8711
Neg Pred Value	0.8595
Prevalence	0.6484
Detection Rate	0.6055
Detection Prevalence	0.6951
Balanced Accuracy	0.8395
'Positive' Class	g

Table 16: RF results on the Test Set

The final results of our model, with several Statistics computed in Table 14, the AUC of 0.86 in Figure 16 and the ROC Curve in Figure 17 demonstrate a good model. For instance, the No Information Rate is the proportion of the most frequent class in the dataset. If the model was simply predicting the most frequent class for all observations,

it would be correct 64.84% of the time. A good model should have an accuracy significantly higher than this rate, which is the case.

The positive predicted value is the precision of the model. It measures the proportion of positive identifications that were actually correct. A high value (87.11%) indicates that the model is highly accurate when it predicts a positive class, and the sensitivity measures the proportion of actual positives correctly identified. A high sensitivity (93.39%) indicates the model is very effective in detecting true positives.

There is a trade-off between this and a higher rate of false positives, as indicated by the lower specificity compared to sensitivity, but we think this is perfectly fine in this context, as explained in Subsection 3.2

3.5 Explainability

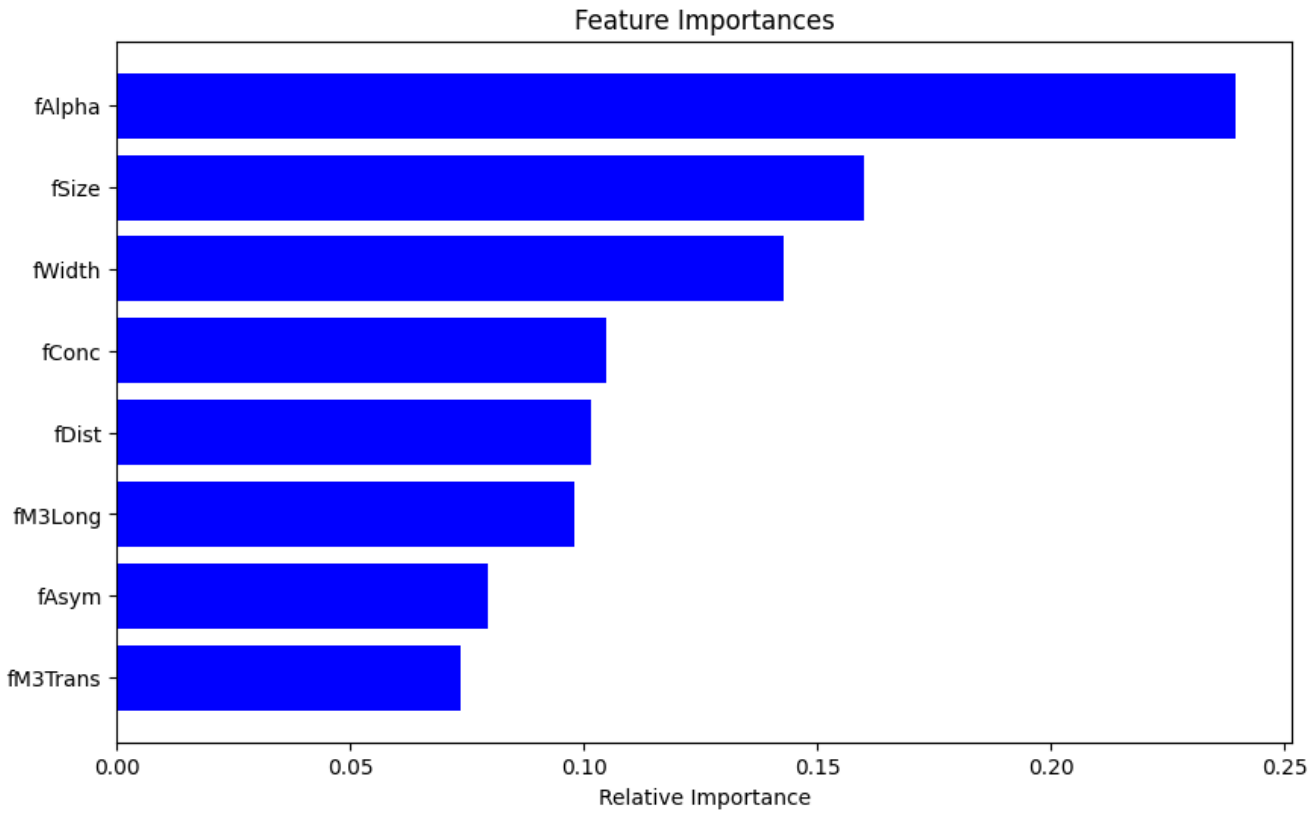

Figure 8: Feature importance for the final Random Forest classifier

In Figure 8 we got the relative importances for each predictor in the final model. The fAlpha feature, which represents the angle of the major axis with a vector to the origin, emerged as the most influential with 0.25.

Next in line are the predictors fSize and fWidth, indicating that the magnitude extent of the observations were also crucial. The importance of fSize suggests that the scale of the phenomena is a strong classifier, while fWidth's significance points to the spatial distribution's role in the identification process.

Orientation, size, and spatial distribution demonstrate being effective for classification. The lesser importance of shape and symmetry characteristics, compared to orientation or size, provides some interesting insights into the nature of noise and gamma signals.

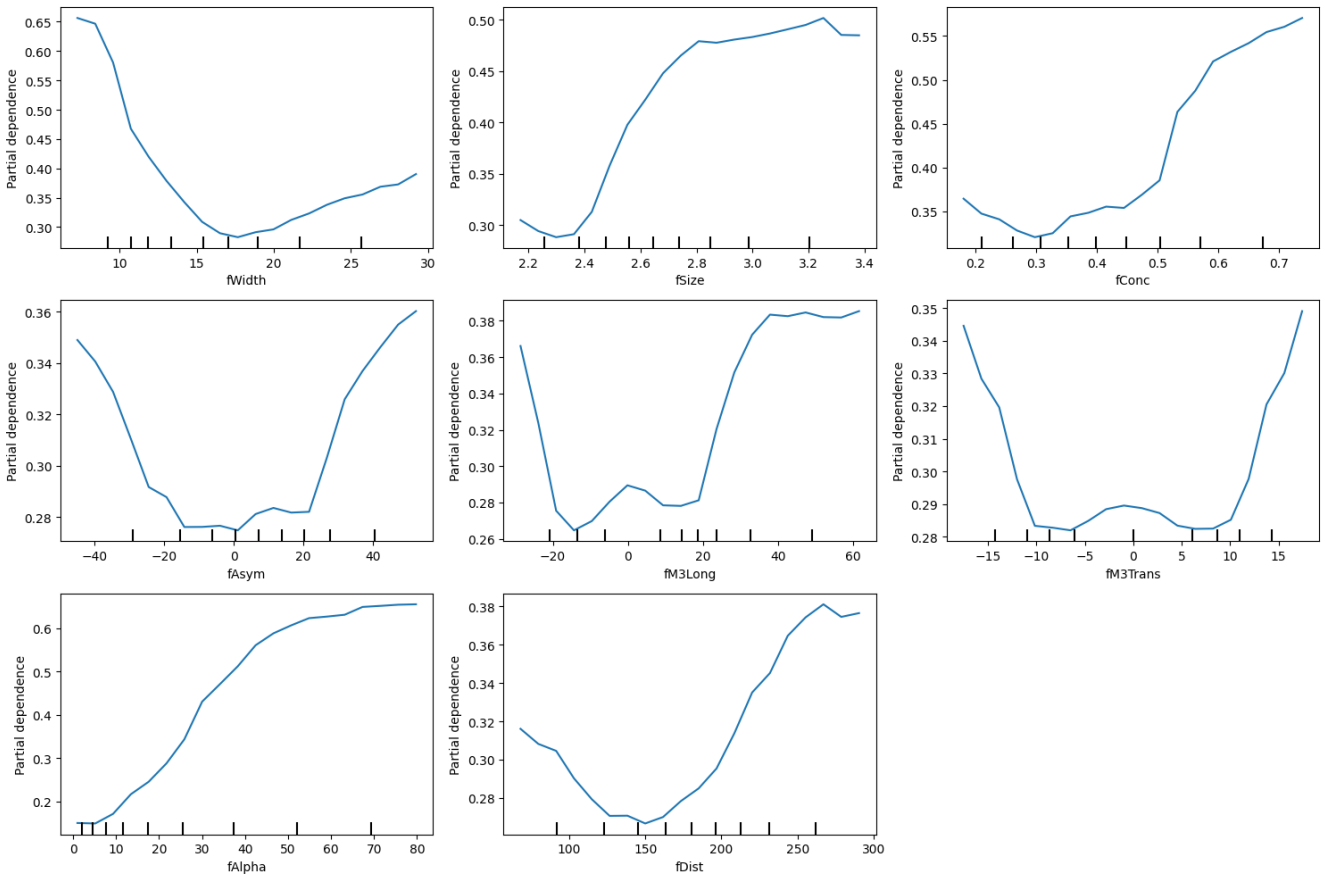

Figure 9: Partial Dependance plot for the final Random Forest classifier

Finally, we reviewed the Partial Dependance plot (Figure 9), that showed some foreseeable results. Predictors like fSize, fConc, fAlpha, fAsym, and fM3Trans showed a typical trend where for values close to 0 the importance of those variables has a minimum, and it ascends when going to the extreme values. For some variables like fWidth, this minimum is close to 17, not 0, while for fm3Long there are local minima around 0.15 and -0.15 Similarly, for fDist the global minima is around 150

This means that beyond some thresholds, some values carry more weight in the decision-making process, and displays that the relationship between the predictors and the importance is non linear.

4 Conclusions

To sum up, this project has given us the opportunity to work with new tools, try new methods seen in this semester and, finally, make some decisions. In this work we provided some data insights and a robust framework for selecting the best method based in our requirements, and we observed that the best results for the purposes of the experiment were obtained using a Random Forest, close to a SVM with RBF Kernel. However, we would chose a Random Forest due to its faster training time.

This method seems to be relying on the fAlpha, fSize and fWidth features mostly. That means that the orientation, size, and spatial distribution are being considered as the most important parts of the data for the classification.

The results on the test set demonstrate a fairly good classifier for the task, capturing the positive class with a high value of precision and sensitivity, without having a remarkably bad specificity.

4.1 Future Work

- Use PCA or other dimensionality reduction techniques to explore other solutions, as calculating cross validation with a hyperparameter grid and Kernel Matrices takes many computational resources.

- Explore different Kernels for SVM methods and Random Forest parameters, as it seems promising.
- Experiment with Neural Networks, as it is state of the art in several research topics.
- Add expert knowledge for more explainability and feature engineering.
- Try SMOTE or other over sampling strategy to increase the number of observations.
- Further investigate in HPC for lowering the training times.

References

[1] R.K. Bock, A. Chilingarian, M. Gaug, F. Hakl, T. Hengstebeck, M. Jirina, J. Klaschka, E. Kotrc, P. Savicky, S. Towers, A. Vaicilius, 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 Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 516(1-2):511–528, 2004.

[2] M. Gaug. Amanda event reconstruction and cut evaluation methods. In *2nd Workshop on Methodical Aspects of Underwater / Ice Neutrino Telescopes*, pages 123–130, Hamburg, August 2001. DESY-PROC-2002-01. August 15-16.

[3] Afshin Gholamy, Vladik Kreinovich, and Olga Kosheleva. Why 70/30 or 80/20 relation between training and testing sets: A pedagogical explanation. Technical Report 1209, Departmental Technical Reports (CS), 2018.

[4] D. Heck et al. CORSIKA, A Monte Carlo code to simulate extensive air showers. *Forschungszentrum Karlsruhe*, FZKA(6019), 1998.

5 Appendix

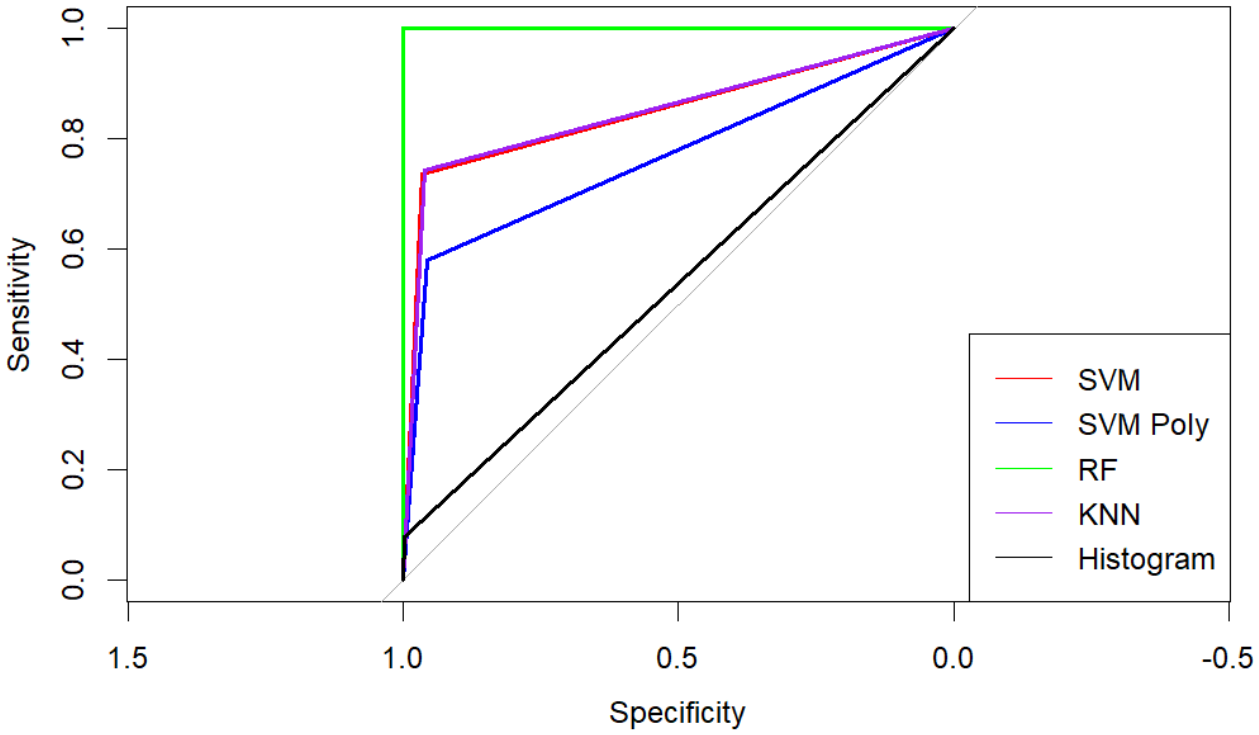

Figure 10: ROC-AUC for the final models on the train set

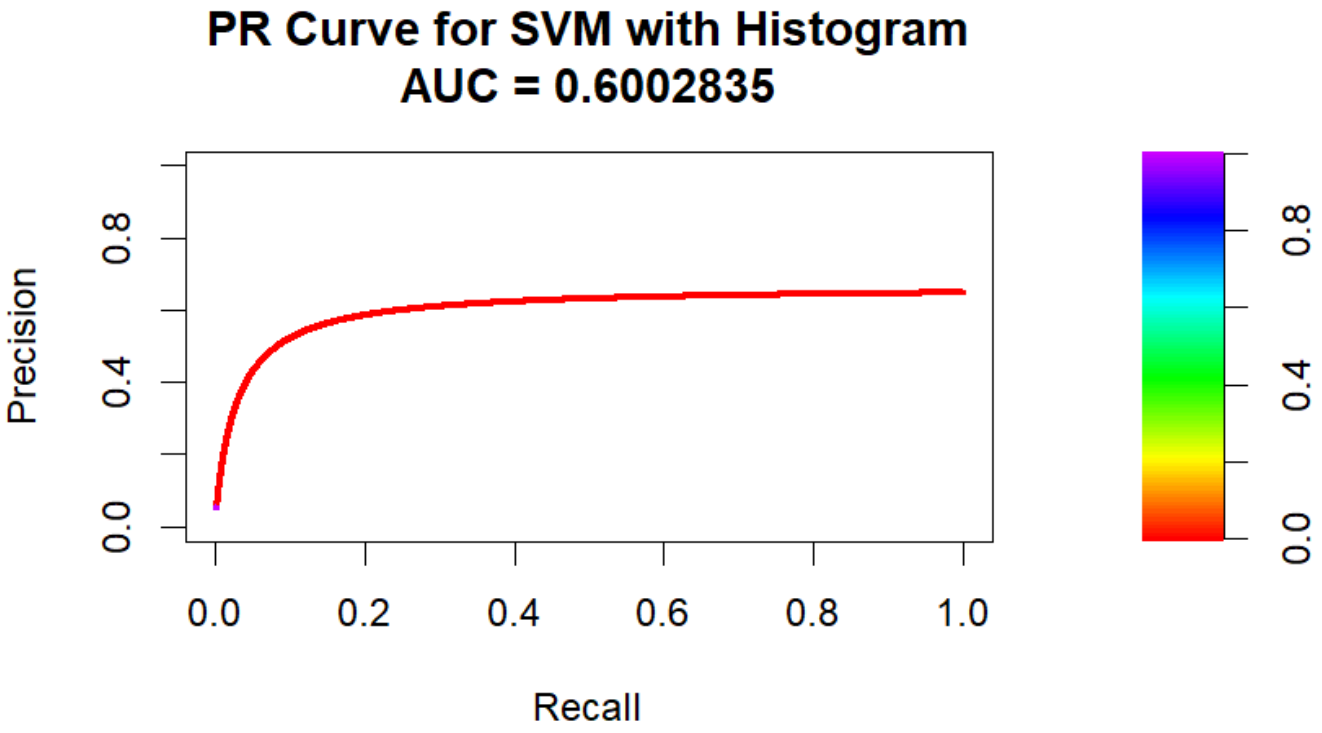

Figure 11: Precision Recall Curve for the SVM with Histogram Kernel

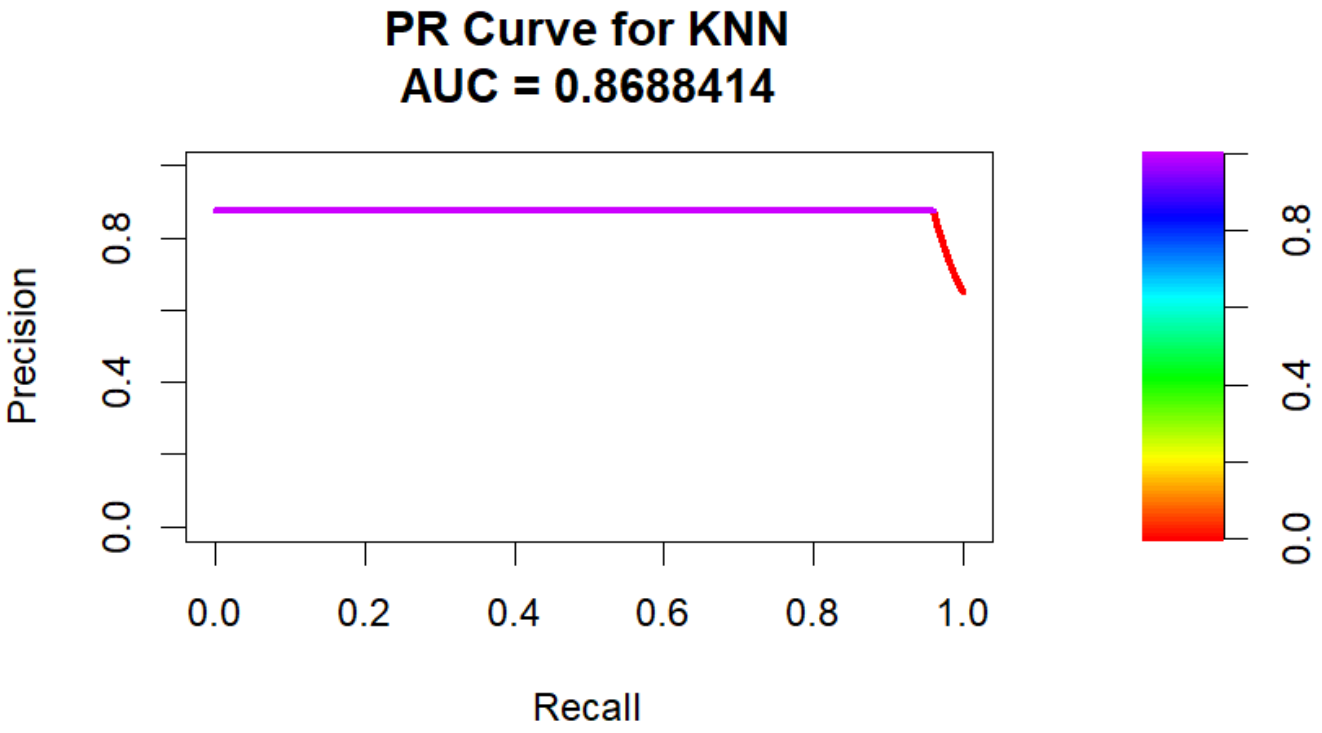

Figure 12: Precision Recall Curve for the KNN

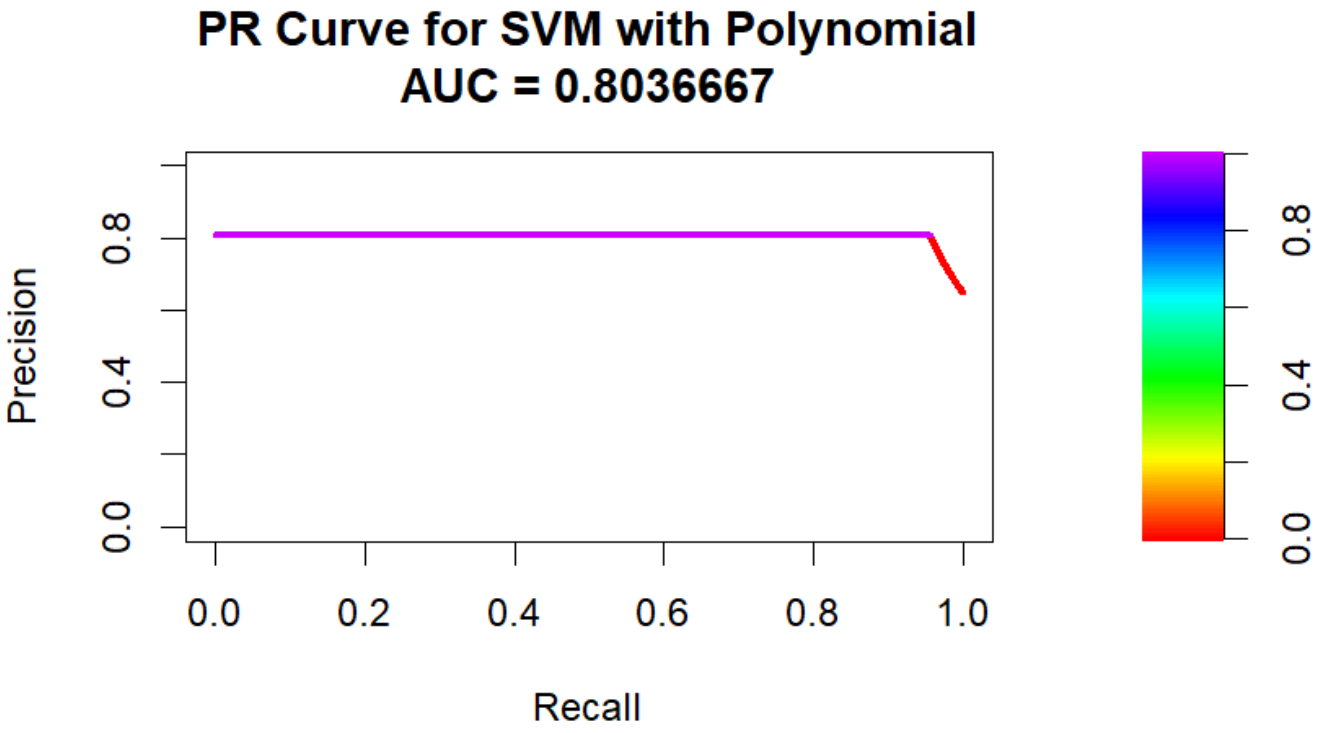

Figure 13: Precision Recall Curve for the SVM with Polynomial Kernel

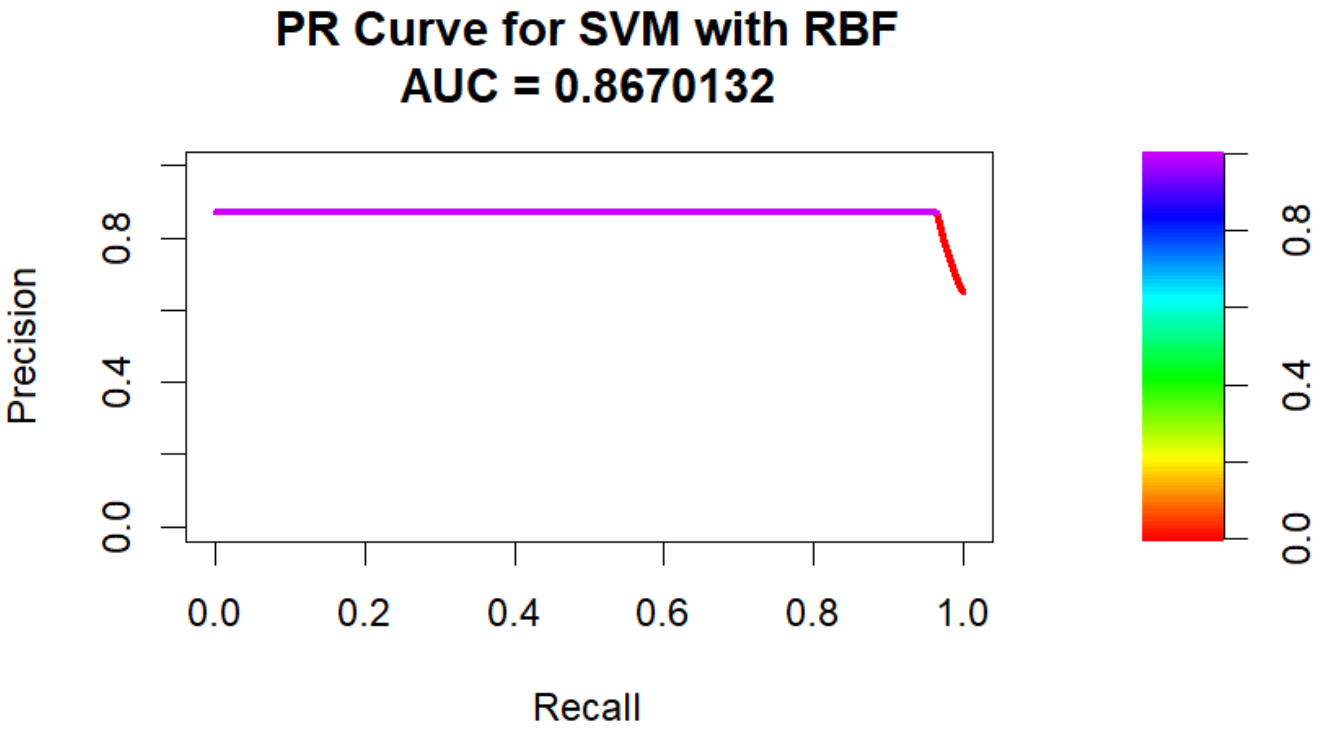

Figure 14: Precision Recall Curve for the SVM with RBF Kernel

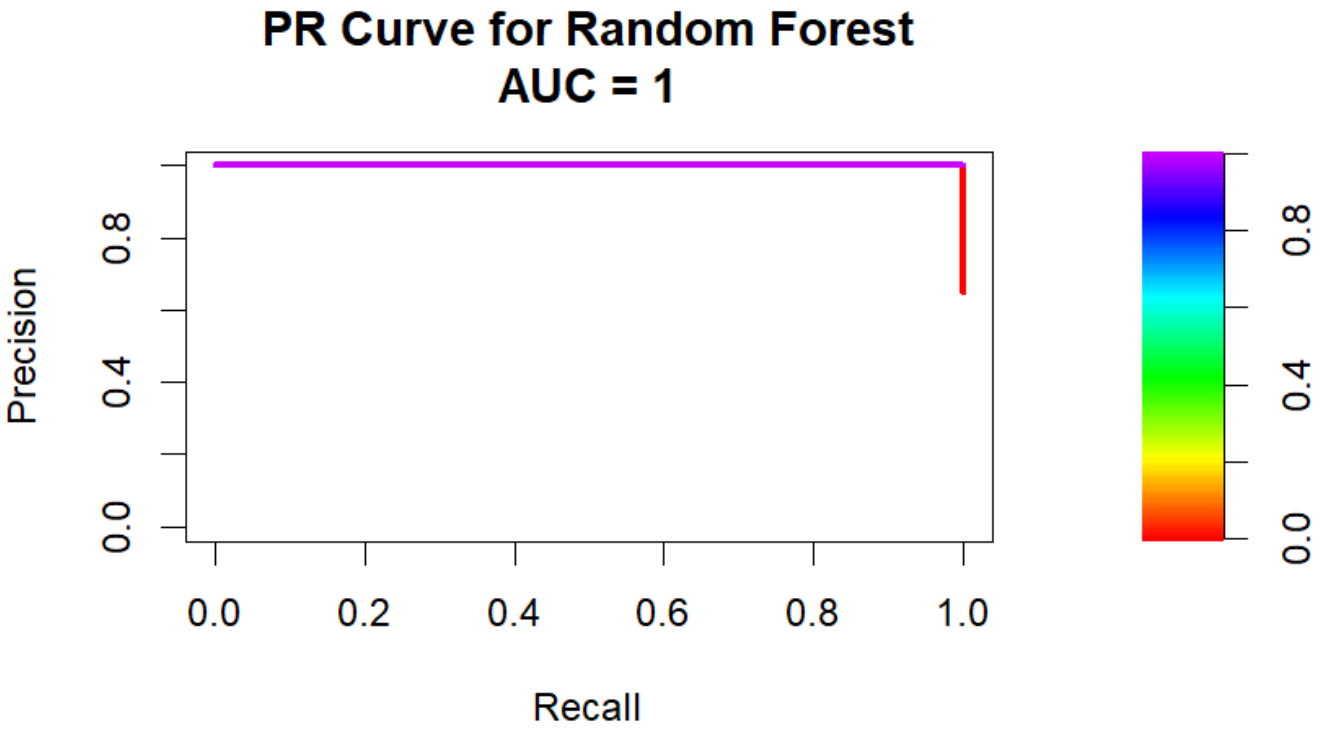

Figure 15: Precision Recall Curve for the Random Forest

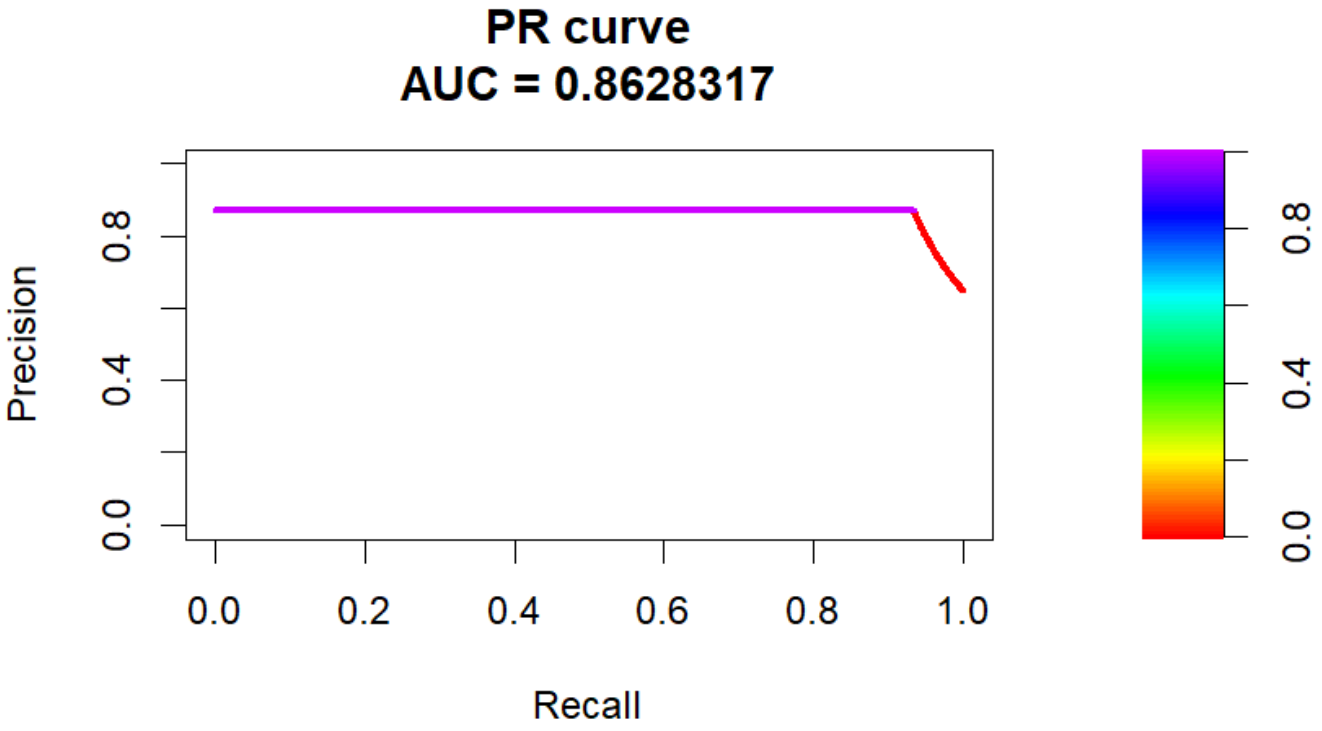

Figure 16: Precision Recall Curve for the Random Forest on the Test Set

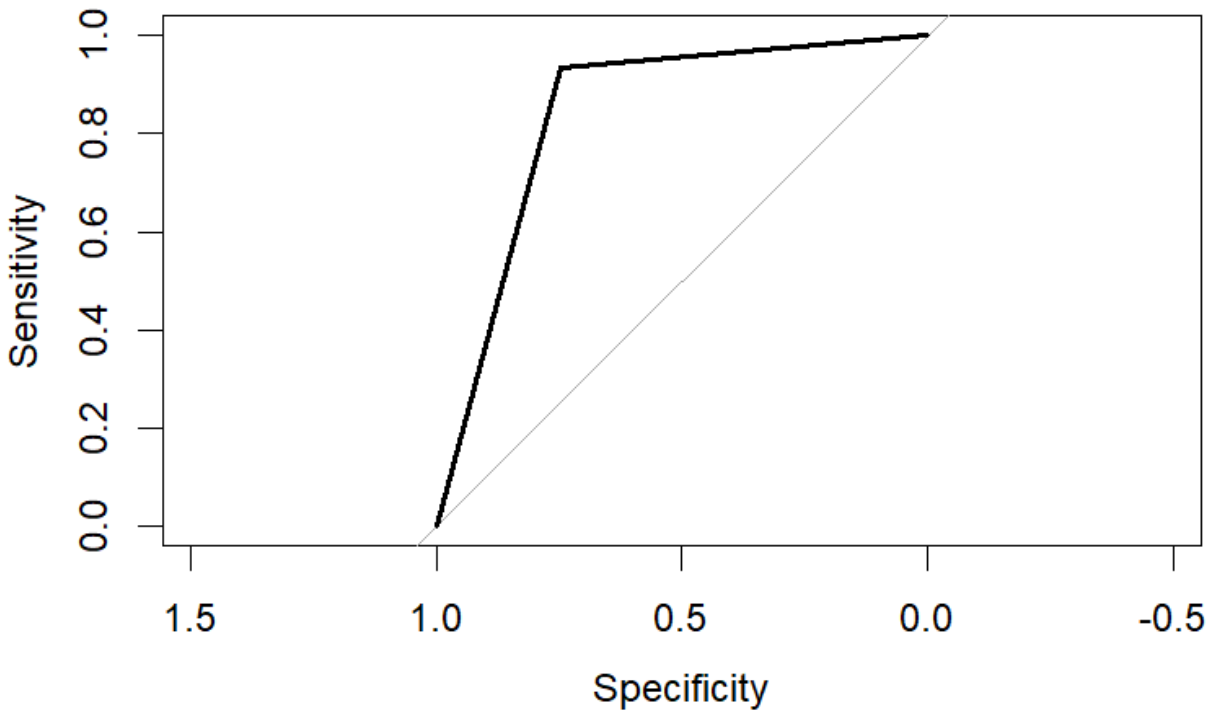

Figure 17: ROC Curve for the Random Forest on the Test Set