Fifty Years of Pulsar Candidate Selection:

Replicating and Testing the Authors' Conclusions on Pulsar Identification Results for the HTRU2 Data Set Using Alternative Data Mining Techniques

Seth Lewis, Taylor Poteete, Kaleigh Roach, Laurence Thompson
Data Science
University of Alabama at Birmingham
Birmingham AL USA

ABSTRACT

Identifying pulsars is a data intensive task that has driven astronomers to seek out new methods of pulsar analysis and classification. *Fifty years of pulsar candidate selection: from simple filters to a new principled real-time classification approach* presented a new approach to quickly classify large numbers of pulsar candidates. Their approach included eight new derived features as well as a tree-based classification algorithm to be used with these features, known as the Gaussian-Hellinger Very Fast Decision Tree (GH-VFDT). In this report, we aimed to both confirm the significance of their eight new features, as well as improve upon the results of their pulsar classification using other classification methods. First, to confirm feature relevance we both clustered the data to experimentally demonstrate the statistically significant difference in feature values for pulsars and non-pulsars, as well as using built in feature importance for our tree-based classification algorithms to determine which features were most important. Second, to improve upon their classification results, we used three different classification algorithms: Random Forest, XGBoost, and AdaBoost. For confirming feature importance, we achieved our best clustering results using Spectral clustering normalized with quantile scalar which yielded a Fowlkes-Mallow score of 97%, and we confirmed the top four most important features listed in this paper - excess kurtosis, mean, and skewness of the integrated profile curve and standard deviation of the DM-SNR curve - to also be the top four most important features of our XGBoost classification model. For improving upon their classification methods, our classification results outperformed those of the paper. Specifically, when using AdaBoost with a 5-fold cross validation data split, compared to the paper's GH-VFDT our classification model achieved equal or higher results in every scoring metric used by this paper.

UAB'2021, April 2021, Birmingham, Alabama USA

CCS CONCEPTS

• Data Science ~ Data Mining • Data Science ~ Data Visualization • Data Science ~ Machine Learning

KEYWORDS

Pulsars, Radio Astronomy, Classification, Clustering, Normalization, Feature Importance, Confusion Matrix, Principal Component Analysis, SHAP, Gradient Boosting, Naïve Bayes', Random Forests, K-Means, Clustering Evaluation, Classification Evaluation, Cross Validation, Grid Search, OPTICS

ACM Reference format:

Seth Lewis, Taylor Poteete, Kaleigh Roach, Laura Thompson. UAB 2021. Replicating and Testing the Authors' Conclusions on Pulsar Identification Results for the HTRU2 Data Set Using Alternative Data Mining Techniques.

1 Introduction

For as long as humans have walked the earth, our eyes have been drawn to the stars. With ever-developing technologies, our ability to learn about the universe outside of our atmosphere continues to grow. One of the most important instruments in the field of astronomy is the radio telescope. Just as a regular telescope allows the observation of light waves in the electromagnetic spectrum, a radio telescope provides the radio frequency section of the electromagnetic spectrum.

A major astronomical discovery made by Jocelyn Bell in 1967 was that of pulsars. What she saw as a flickering light, we now know much more about. Pulsars are now understood to be compact, highly magnetized stars that emit beams of radiation from its magnetic poles [1][1]. These beams of radiation vary in frequency and pulsars have since been discovered that emit radio, visible light, gamma, and

[1] Cofield, Calla. 2016. What Are Pulsars? (April 2016). Retrieved April 20, 2021 from https://www.space.com/32661-pulsars.html

x-ray frequencies. Pulsars rotate extremely fast and in very precise intervals that range from seconds to milliseconds. The "pulses" that are observed are detected as the pole emits radiation directed at the earth. For example, consider a pulsar like what Jocelyn Bell observed, a pulsar that emits visible light waves with a 1.33 second rotation interval would be seen as a flash of light every 1.33 seconds.

Radio telescopes allow the study of pulsars that emit radiation on a wide range of frequencies on the electromagnetic spectrum. The study of pulsars goes hand in hand with the search for pulsars. Astronomers have been developing methods to identify pulsars faster so that more resources can be applied to studying the confirmed pulsars as opposed to finding them.

To expedite the discovery of pulsars, the boundaries of radio telescopes are being pushed. An intergovernmental radio telescope project, Square Kilometer Array (SKA), is being developed. SKA implementation is planned for mid 2020-2030 and will have radio telescope sites across Australia and South Africa. As the name suggests, the goal is to maintain a data collection area of one square kilometer. The SKA will be able to produce data that is 50 times more sensitive than any other radio instrument [2][2].

With this massive amount of increasingly detailed data, existing methods used to classify pulsars will not be able to keep pace with the data that the next generation of telescopes will provide. *Fifty years of pulsar candidate selection: from simple filters to a new principled real-time classification approach* proposes the use of classification algorithms in real time for pulsar identification and presents a new set of features and data to be used for classification. The paper also specifically presents a purpose-built tree-based machine learning classifier, known as the Gaussian Hellinger Very Fast Decision Tree, although results were given for multiple classification algorithms. Their goal was to create a set of features and training data that would scale well with increasing pulsar candidate numbers and the noise that comes from larger amounts of data.

This report focuses on confirming the significance of their features as well as performing additional classification methods on the training data in hopes of improving the results of the paper. Section 2 will discuss the problem basis and results for *Fifty years of pulsar candidate selection: from simple filters to a new principled real-time classification approach* [3][3]. Section 3 will explain how the provided data set was prepared for classification and clustering. In section 4 we will demonstrate how we clustered the data and statistically determined feature redundancy as well as the classification methods we used on the data. In section 5 we will provide our results and discuss how they compared to the methods used in [3]. All tests were performed in Python and recorded in the form of Jupyter Notebooks.

2 Previous Work and the Problem Basis

In recent years, our ability to identify potential pulsar candidates has greatly improved, and radio telescopes now regularly find tens of millions of potential pulsar candidates in a single survey. Additionally, the Square Kilometer Array described in the previous section is predicted to find candidate numbers much higher than this, potentially finding 6 million candidates in a single observation and billions of candidates over the course of an entire survey [3]. Unfortunately, most of the potential pulsar candidates found are determined to be radio noise, and with millions of potential candidates, it is crucial to find ways to quickly filter candidates and determine those that are likely to be pulsars.

In response to this problem, Lyons et al. proposed a pulsar classification method that could be used online and in real-time to quickly filter the input data acquired by radio telescopes. Their proposed solution includes a new set of features and an accompanying set of labeled pulsar candidates [4] as well as a purpose-built tree-based machine learning classifier, known as the Gaussian Hellinger Very Fast Decision Tree which was developed by the same research group two years earlier in 2014 [5]. The data used for their work includes three different sets of pulsar candidates summarized in table 5 from [3] shown below. Our project uses the data set HTRU2 because the other two data sets were not available to us.

Table 1. Pulsar candidate data sets used. [3, Tab. 5]

Data set	Examples	Non-pulsars	Pulsars
HTRU 1	91 192	89 995	1196
HTRU 2	17 898	16 259	1639
LOTAAS 1	5053	4987	66

2.1 Feature Selection

The Lyons et al. paper [3] specifically focuses on the development of a set of new features which could be used to classify pulsar candidates. The paper acknowledges that while there are already existing methods for classifying pulsar candidates using machine learning techniques, the features used in these methods are often insufficient. Often features are selected from heuristics used in manual pulsar

[2] "Facts and figures". SKA Organization. Archived from the original on 28 July 2012. Retrieved 26 May 2012.

[3] R. J. Lyon, B. W. Stappers, S. Cooper, J. M. Brooke, J. D. Knowles, Fifty Years of Pulsar Candidate Selection: From simple filters to a new principled real time classification approach, Monthly Notices of the Royal Astronomical Society 459 (1), 1104-1123, DOI: 10.1093/mnras/stw656

candidate selection which subjects them to human bias. Other times, features are not clearly defined, or it is unclear how a feature was derived from the raw pulsar data. This makes it difficult to reproduce and build upon the work of others. As a result, Lyons et al. sought to develop a set of features that would

"(i) minimize biases and selection effects,
(ii) be survey independent for data interoperability,
(iii) be implementation independent, with concise mathematical
definitions allowing for reproducibility,
(iv) be evaluated using a statistical framework that enables comparison and reproducibility,
(v) guard against high dimensionality,
(vi) be accompanied by public feature generation code, to facilitate co-operation and feature improvement,
(vii) be supplied in a standard data format,
(viii) be evaluated on multiple data sets to ensure robustness."

Using these guidelines, they developed a set of eight features that were mathematically well defined, highly reproducible, and easily evaluated allowing them to be quickly determined from input data of a radio telescope. These features are outlined in table 4 from [3], shown below.

Table 2. Definitions for the eight features derived from the integrated pulse profile P = {p1, … , pn}, and the DM-SNR curve D = {d1, … , dn}. [3, Tab. 4]

Feature	Description	Definition
Prof.μ	Mean of the integrated profile P.	$\frac{1}{n}\sum_{i=1}^{n} p_i$
Prof.σ	Standard deviation of the integrated profile P.	$\sqrt{\frac{\sum_{i=1}^{n}(p_i-\bar{P})^2}{n-1}}$
Prof.k	Excess kurtosis of the integrated profile P.	$\frac{\frac{1}{n}(\sum_{i=1}^{n}(p_i-\bar{P})^4)}{(\frac{1}{n}(\sum_{i=1}^{n}(p_i-\bar{P})^2))^2}-3$
Prof.s	Skewness of the integrated profile P.	$\frac{\frac{1}{n}\sum_{i=1}^{n}(p_i-\bar{P})^3}{\left(\sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i-\bar{P})^2}\right)^3}$
DM$_\mu$	Mean of the DM-SNR curve D.	$\frac{1}{n}\sum_{i=1}^{n} d_i$
DM$_\sigma$	Standard deviation of the DM-SNR curve D.	$\sqrt{\frac{\sum_{i=1}^{n}(d_i-\bar{D})^2}{n-1}}$
DM$_k$	Excess kurtosis of the DM-SNR curve D.	$\frac{\frac{1}{n}(\sum_{i=1}^{n}(d_i-\bar{D})^4)}{(\frac{1}{n}(\sum_{i=1}^{n}(d_i-\bar{D})^2))^2}-3$
DM$_s$	Skewness of the DM-SNR curve D.	$\frac{\frac{1}{n}\sum_{i=1}^{n}(d_i-\bar{D})^3}{\left(\sqrt{\frac{1}{n}\sum_{i=1}^{n}(d_i-\bar{D})^2}\right)^3}$

Before classifying the pulsar data, Lyons et al. used several statistical methods to determine the potential importance of each feature. First, they tested whether the difference between pulsars and non-pulsars was statistically significant. To do so, they first normalized the distributions of each feature on the range [0, 1] and centered the data about the non-pulsar median, resulting in the box plot figure 6 from [3] below, which shows that there is a clear difference between the distributions for pulsar and non-pulsar candidates. Lyons et al. then applied a two-tailed student's t-test to the feature data and found that for all data sets, there was a statistically significant difference between the pulsar and non-pulsar distributions at α = 0.01. In our own project, we attempted to experimentally demonstrate the statistically significant differences in the feature distributions by clustering the data to determine if there was a clear difference between the two. The details and results of this will be discussed in sections 3, 4 and 5.

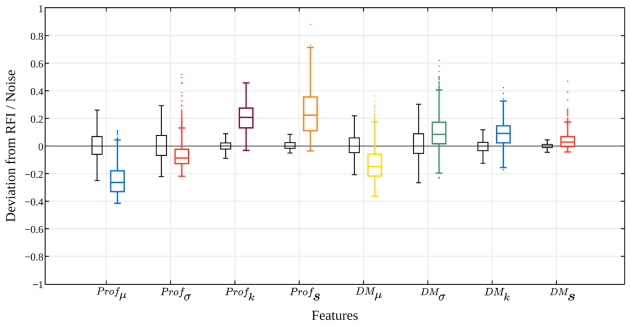

Figure 1. Box plots showing the linear separability of features used. [3, Fig. 6]

Finally, the most relevant features were ranked using their joint mutual information (JMI) criterion. JMI uses the mutual information (MI) measure which determines how much features reduce uncertainty relative to one another, while considering information redundancy. JMI can be mathematically defined as:

$$\mathrm{JMI}(X^j) = \sum_{X^k \in F} I(X^j X^k; Y)$$

where

$$I(X^j; Y) = H(X^j) - H(X^j|Y),$$

$$H(X^j) = -\sum_{x \in X^j} P(x)\log_2 P(x)$$

and

$$H(X^j|Y) = -\sum_{y \in Y} p(y) \sum_{x \in X^j} P(x|y)\log_2 P(x|y)$$

, where X^j is a feature and Y is a class label, $H(X^j)$ is the entropy of a feature, and $I(X^j; Y)$ is the mutual information. The results of this are shown in table 8 of [3] below.

Table 3. JMI rank of each feature, ranked according to average JMI across all three data sets. Lower scores are better. [3, Tab. 8]

Feature	Dataset			Avg. rank
	HTRU 1	HTRU 2	LOTAAS 1	
Prof.k	1	1	1	1
Prof.μ	3	3	3	3
DM$_\sigma$	2	2	8	4
Prof.s	4	4	6	4.7
DM$_k$	6	6	2	4.7
Prof.σ	7	5	5	5.7
DM$_\mu$	5	7	7	6.4
DM$_s$	8	8	4	6.7

We attempted to confirm their feature importance which we will be reporting in sections 3 and 4 then discussing in section 5.

2.2 Classification Results

Finally, the paper uses their new features to classify the pulsar data using their Gaussian-Hellinger Very Fast Decision Tree (GH-VFDT) algorithm and four other classification methods: the decision tree algorithm C4.5, MLP neural network, a simple probabilistic classifier Naive Bayes (NB), and the standard linear soft-margin support vector machine (SVM). The models were trained using a split containing 200 pulsars and 200 non-pulsars with the rest of the data being used as test data. They then determined the performance of these classification models using the metrics outlined in table 10 of [3] below.

Table 4. Evaluation metrics used for classifier performance. [3, Tab. 10]

Statistic	Description	Definition
Accuracy	Measure of overall classification accuracy.	$\frac{(TP+TN)}{(TP+FP+FN+TN)}$
False positive rate (FPR)	Fraction of negative instances incorrectly labelled positive.	$\frac{FP}{(FP+TN)}$
G-Mean	Imbalanced data metric describing the ratio between positive and negative accuracy.	$\sqrt{\frac{TP}{TP+FN} \times \frac{TN}{TN+FP}}$
Precision	Fraction of retrieved instances that are positive.	$\frac{TP}{(TP+FP)}$
Recall	Fraction of positive instances that are retrieved.	$\frac{TP}{(TP+FN)}$
F-Score	Measure of accuracy that considers both precision and recall.	$2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$
Specificity	Fraction of negatives correctly identified as such.	$\frac{TN}{(FP+TN)}$

Using these metrics, they acquired the following results, where darker colors indicate higher scores.

Table 5. Final classification results for the five classification algorithms used. This table has been altered such that darker colors indicate higher values. Data adapted from [3, Tab. 11]

Paper Classification Results

	Data Set	Algorithm	G-Mean	F-Score	Recall	Precision	Specificity	FPR	Accuracy	AUC
0	HTRU2	C4.5	0.926000	0.740000	0.904000	0.635000	0.949000	0.051000	0.946000	-
1		MLP	0.931000	0.752000	0.913000	0.650000	0.950000	0.050000	0.947000	-
2		NB	0.902000	0.692000	0.863000	0.579000	0.943000	0.057000	0.937000	-
3		SVM	0.919000	0.789000	0.871000	0.723000	0.969000	0.031000	0.961000	-
4		GH-VFDT	0.907000	0.862000	0.829000	0.899000	0.992000	0.008000	0.978000	-

In summary, the Lyons et al. paper [3] seeks to provide a quick and efficient way to classify large numbers of potential pulsar candidates online and in real-time. Their presented solution was a new set of features that could be applied to any pulsar data as well as a classification algorithm, the Gaussian-Hellinger Very Fast Decision Tree to use with their provided features and data. They then compared the performance of this algorithm to other classification algorithms. All algorithms performed well, indicating that their features are relevant, and their GH-VFDT performed especially well, providing the best results for F-score, precision, specificity, and accuracy.

Our project seeks to accomplish two main goals. First, to experimentally confirm the significance of the features presented in paper [3], which we accomplished both by clustering the data and by using built in feature importance methods for our classification models. Second, to attempt to improve upon the classification results of [3]. To do so, we classified the provided data using three tree-based algorithms not used in [3] - Random Forest, XGBoost, and AdaBoost, as well as several different methods for splitting the training data in hopes of achieving better results than those achieved by the paper. We also used the Naive Bayes classifier which was used by the paper to confirm their results.

3 Data Collection and Preparation

Our data was provided by the authors of the cited paper [3, 4][4] and contained only continuous numerical features with no missing data and therefore no cleaning was needed. Since no nominal or ordinal values were included, no re-encoding was needed as well. It should be noted that the data included no positional information for the pulsar candidates, all the features were collected by analyzing radio frequency emissions from candidate signals and the ground truth was determined manually by astronomers either as known pulsars or confirmed by observation and calculation. The data contained 17898 instances across 8 features and 1 binary class. The binary class counted

[4] R. J. Lyon (2016): HTRU2. figshare. Dataset. https://doi.org/10.6084/m9.figshare.3080389.v1

16259 confirmed noise/non-pulsars and 1639 confirmed pulsars.

Upon pulling the data into our code as pandas data frames and labeling the columns to match the features extracted by the authors [3], we proceeded to split our data. This was done separately for the purposes of exploration through clustering and for the results replication via classification.

Table 6. Head of data after import into code. This table includes the ground truth column that was separated for training and testing in later steps.

Data Head

	mean_IP	standDev_IP	excessKurt_IP	skewness_IP	mean_DMSNR	standDev_DMSNR	excessKurt_DMSNR	skewness_DMSNR	pulsar
0	102.507812	58.882430	0.465318	-0.515088	1.677258	14.860146	10.576487	127.393580	0
1	103.015625	39.341649	0.323328	1.051164	3.121237	21.744669	7.735822	63.171909	0
2	136.750000	57.178449	-0.068415	-0.636238	3.642977	20.959280	6.896499	53.593661	0
3	88.726562	40.672225	0.600866	1.123492	1.178930	11.468720	14.269573	252.567306	0
4	93.570312	46.698114	0.531905	0.416721	1.636288	14.545074	10.621748	131.394004	0

Table 7. Description of data statistics for all 9 columns.

Data Description

	mean_IP	standDev_IP	excessKurt_IP	skewness_IP	mean_DMSNR	standDev_DMSNR	excessKurt_DMSNR	skewness_DMSNR	pulsar
count	17897.000000	17897.000000	17897.000000	17897.000000	17897.000000	17897.000000	17897.000000	17897.000000	17897.000000
mean	111.078321	46.549021	0.477897	1.770417	12.614926	26.326918	8.303574	104.859419	0.091580
std	25.652705	6.843040	1.064056	6.168058	29.473637	19.471042	4.506217	106.517270	0.288440
min	5.812500	24.772042	-1.876011	-1.791886	0.213211	7.370432	-3.139270	-1.976976	0.000000
25%	100.929688	42.375426	0.027108	-0.188528	1.923077	14.437330	5.781485	34.957119	0.000000
50%	115.078125	46.946435	0.223241	0.198736	2.801839	18.459977	8.433872	83.068996	0.000000
75%	127.085938	51.022887	0.473349	0.928206	5.464883	28.428152	10.702973	139.310905	0.000000
max	192.617188	98.778911	8.069522	68.101622	223.392141	110.642211	34.539844	1191.000837	1.000000

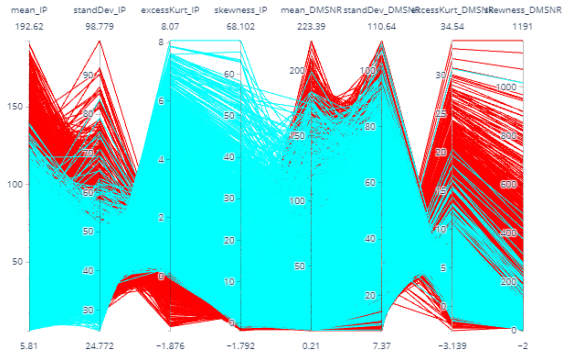

Figure 2. Raw data visualized as parallel coordinates across all features and ground truth in cyan.

3.1 Splitting for Clustering, Cross Validation, Training, and Testing

For clustering the data was broken into two sets, features, and ground truth. These were labeled as pulsar_data and ground_truth respectively. For classification a series of training and test splits were created to accommodate the author's method; 3, 5, and 10-fold cross validation, and an experimental split based upon the author's method but with a larger training sample (this was only used on the final stage of classification using XGBoost). The training and test splits for the cross validation match the percentages of the folds.

It was particularly important to stratify our splits (See section 4.1) to accommodate cross validation due to the imbalance of negative to positive classes in the ground truth. To emulate the authors' methods, the pulsar data was sorted, and 400 samples were selected for training, 200 negative and 200 positives, as X_train and the related ground truth as y_train. The remaining data was used as the testing splits broken into X_test and y_test. Both the training and testing data were shuffled before final splitting to improve modeling. This was done before final splitting to prevent the ground truth labels from getting separated from their appropriate samples. Lastly a single experimental split was created and used during the XGBoost classification. Like the author's it was a 50/50 split of positive to negative classes of size 2000 where 1000 were negative and 1000 were positive classes. Again, the remaining data was used for the test splits. **It should be noted that, for any calculation, the ground truth column has not been included for any purpose except evaluation of results.**

3.2 Normalization for Clustering

During our clustering multiple means of normalization were performed and tested against the raw data. Most used the z-score method; but robust scalar, power transformer, and quantile scaling (against a gaussian distribution) were also tested from the scikit-learn implementations. Our best results were achieved with z-score and quantile scaling (quantile specifically for spectral clustering). Z-score was used due to the floating-point nature of the raw data since this type of normalization is usually sufficient when continuous numerical data of this type is provided. The equation for calculating z-score appears below:

Equation 1. z-score normalization formula.

$$Z = \frac{x - \mu}{\sigma}$$

Quantile, Power and Robust were selected to see if their ability to handle marginal outliers would improve our results. Only Quantile improved scoring over z-score in a single instance so we will not be defining the other methods (Robust and Power) here.

The quantile function as used within scikit-learn takes any input value and maps it to the range p in (0,1) using a normalization function input to the quantile function. In our case, having the gaussian argument passed into the function allows the algorithm to use the inverse of the cumulative distribution function to calculate the new values. The definition of this formula appears below:

Equation 2. Inverse of the cumulative distribution function used by the quantile scalar function.

$$\Phi^{-1} = \sqrt{2}erf^{-1}(2p-1), p \in (0,1)$$

Normalization was tested but gave no differing results on classification from that of the raw data set and was therefore not included in this report being redundant.

3.3 Principal Component Analysis

Since our data set was of higher dimensionality, Linear PCA, a method of feature dimension reduction, was used to assist in our visualizations. Principal Components Analysis (PCA) is an unsupervised dimensionality reduction technique that constructs relevant features/variables through linear (linear PCA) or nonlinear (kernel PCA) combinations of the original variables (features). Linear transformation of correlated features using dot product to acquire a reduced set of non-correlated features was used in our case.

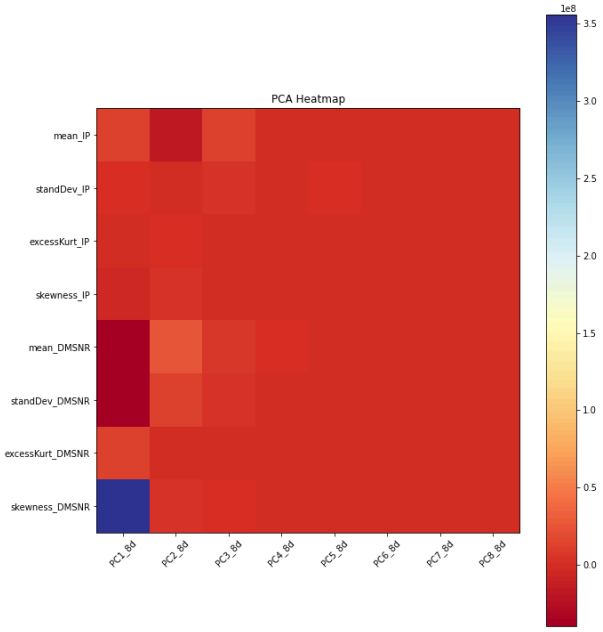

Figure 3. Heat map showing feature importance to PCA dimension reduction before normalization.

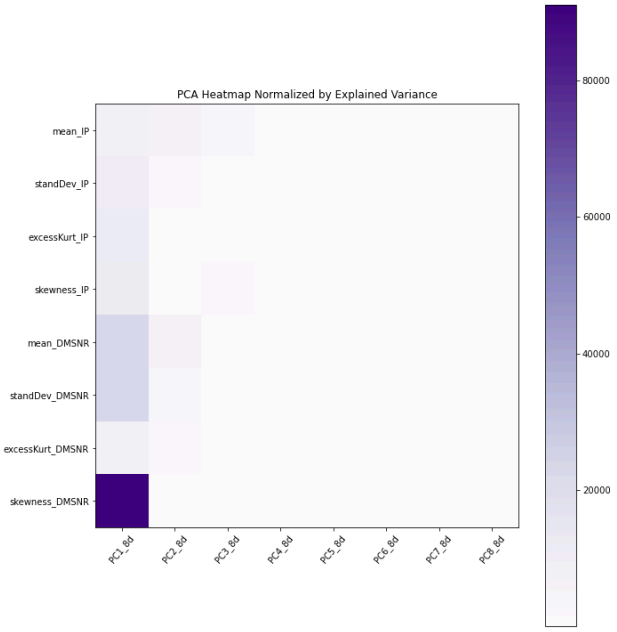

Figure 4. Heat map showing feature importance to PCA dimension reduction after normalization and dot product with explained variance.

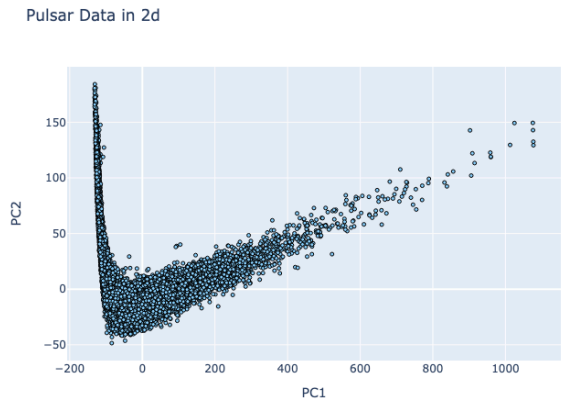

Figure 5. 2D raw data visualization created through PCA dimension reduction (does not include ground truth).

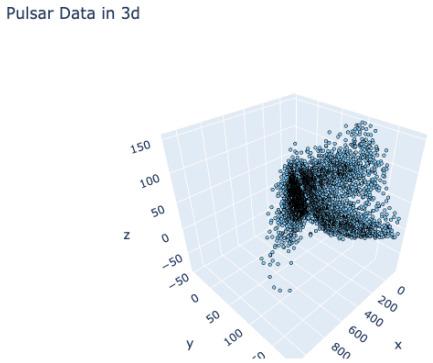

Figure 6. 3D raw data visualization created through PCA dimension reduction (does not include ground truth).

3.4 Correlation Mapping and Initial Feature Importance Testing

Feature correlation is a way to understand the relationships between features in a data set. Correlation can help in predicting values from one another, sometimes it can be used to approximate missing values. Correlation can lead to multicollinearity, a problem that can negatively impact a model. When a predictor variable can be linearly predicted from the other predictor variables it can lead to skewed results. Fortunately, most of the algorithms picked by the authors and us are resistant to this problem. We explore correlation in this section because it is relevant to the authors' documentation.

Feature importance can refer to any number of techniques that assign a score to predictor variables indicating how useful they are in predicting an instance's inclusion in a class. This is central to our verification of the paper's conclusions.

The author's conclusions on the features selected for identifying pulsars included an identification (in list format, see Section 2 or 4.6 for details, tables 3 or x) of feature importance and statement of possible redundancy reduction by features in the dataset, this was tested by us in three separate phases; an initial correlation mapping plus a violin plot of the features, using the built in feature importance of the scikit-learn Random Forest implementation then plotting the results, and again during the XGBoost classification utilizing SHAP or SHapley Additive exPlanations. SHAP will be explained during the report of our classification results (See section 4.6). The correlation mapping showed four sets of highly correlated features and the violin plot gave us a baseline comparison to the paper's feature importance conclusions before we performed any actual clustering or classification.

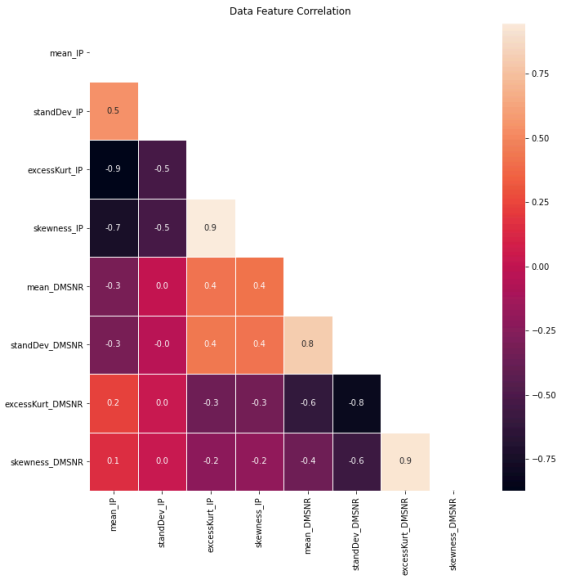

Figure 7. Correlation map showing four pairs of high correlation among the 8 features.

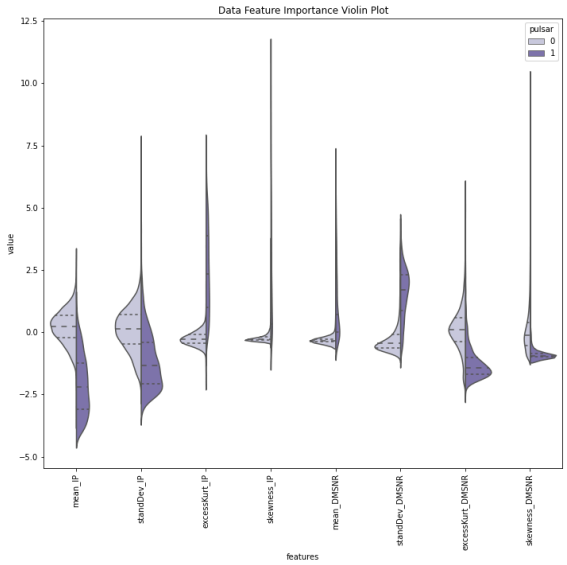

Figure 8. Violin plot indicating that mean_IP, standDev_IP, and excessKurt_DMSNR would be highly influential in positively classifying pulsars.

The outcomes of these preparations will be provided in the following sections. Since our work was an attempt to test and replicate the conclusions of the original paper our methodology followed what we believed was the author's methodology. This was done to the best of our abilities within the limitations of the course structure and with lack of access to additional pulsar data sets using the same feature list.

4 Prediction Models and Results

Many different measures and evaluation metrics were used to analyze the results of our modeling techniques. For completeness we will briefly define the component parts of these metrics and the formulation of the metrics themselves. Two main metrics were the focus of our experimentation Fowlkes-Mallow Index and F-score (F-measure). The authors also evaluated on metrics that make up or contain elements of these two, along with overall accuracy. Both Fowlkes-Mallow and F-Measure utilize TP; True Positives (the predictions that correctly match the ground truth's positive class), TN; True Negatives (the predictions that correctly match the ground truth's negative class), FP; False Positives (the predictions that are incorrectly matched to the ground truth's positive class), and FN; False Negatives (the predictions that are incorrectly matched to the ground truth's negative class). Together these can be turned into a plot called a confusion matrix, an often-useful tool for quickly estimating a classification or clustering model's effectiveness before applying proper scoring calculations. A table representation appears below:

Table 8. Example of confusion matrix.

Confusion Matrix Example

	N/P	Negative	Positive
0	Negative	TN	FP
1	Positive	FN	TP

In addition to FM for clustering we also tested Rand Index (also known as Accuracy), but again focused on FM for evaluating our data exploration even though Rand uses a similar calculation. A perfect clustering result would have a value of 1.0 on this index. Rand Index is defined as:

Equation 3. Rand Index or Accuracy equation.

$$RI = \frac{TP + TN}{TP + FP + FN + TN}$$

The Fowlkes-Mallow Index and G-Mean are the Geometric Mean of True Positives and since we want to maximize the TP for any pulsar search it is a very important metric. Both are rated with 1.0 being a perfect score. Below are FM and G-Mean expanded to their full definition and as their relationship to precision and recall:

Equation 4. Fowlkes-Mallow Index

$$FMI = \sqrt{\frac{TP}{TP + FP} \cdot \frac{TP}{TP + FN}}$$

Equation 5. Geometric mean equation, an alternative definition of Fowlkes-Mallow Index

$$GM = \sqrt{Precision \cdot Recall}$$

Recall is used as a component of the above but is useful on its own as the positive predictive rate, calculated by the formula below:

Equation 6. Recall

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

Precision like recall is another part of the above calculations but is also known as the true positive rate, calculated by the formula below:

Equation 7. Precision

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

The authors of the paper [3] also focused heavily on Specificity, FPR, and Accuracy. FPR is a useful measure for us as well since it tells us how many instances of the negative ground truth class are getting mis-classified by our model and increasing the error of our predictions. A lower FPR is better with 0 being perfect, its calculation is simple and is listed here:

Equation 8. False positivity rate

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

Specificity though important does not tell us as much about our results since the data set was heavily weighted to the negative class of non-pulsars and because it is 1-FPR, but we included it in our results to maintain consistency with the authors' methodology, it is calculated by the following formula:

Equation 9. Specificity or 1-FPR

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

For our classification models F-Score (F-Measure) was our most focused upon scoring metric and was used by our cross validation and grid search implementations to determine the best splits and best hyper parameters for training. We focused in this way because it was an efficient method for limiting the number of FP and FN that could add or subtract from the predictive TP results lessening the effectiveness of our models. The calculation of F-Measure gives high weight to precision and recall. It is also the harmonic mean of these measures. We used F1 because we wanted to value the component measures equally. The formula for F1-Measure appears here:

Equation 10. F1 measure, typical version of F-Score giving equal weight to Precision and Recall.

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

AUC or roc_auc score is the area under the probability curve created from a model. This metric was not used by the authors and was used by us as an initial gauge whether we were on the right track in our classification. It's use in our CV was supplanted by F1 and we will not go into more detail here because it does not factor into the conclusion comparisons or discussion.

4.1 Data Mining Techniques

An overview of a few of the data mining techniques used in our testing will help round out the explanation of our results. For our initial exploration of the data and to gain preliminary understanding on the effectiveness of the authors' provided features we performed several rounds of clustering.

Clustering grouped the instances of the data set based on their similarity, difference from one another, or via a distance metric into separate clusters. Normalization was necessary for clustering and was utilized to prevent marginal outliers from overly skewing the inter and intra cluster distances (See section 3.3). From this we were able to infer if it was worthwhile to proceed with further study and if we were on the right track to replicate, albeit in differing ways, the results of the paper as tested on the HTRU2 pulsar data. Our results should justify our continuation into Classification which was the crux and purpose of the feature set developed by the authors'.

Our classification methods all attempted binary classification whereby multiple predictive models were trained to classify instances of the features in a testing set into the two categorical classes pulsars and non-pulsars.

Grid search, a helper method to test an exhaustive set of hyperparameter combinations, was used when training all models. Along with this Cross validation, a method by which the training data is further split and tested exhaustively across many fractional divisions against a final fraction of evaluation data from the training set and based on some scoring metric (ours was F1-measure as stated above) was used to improve the efficacy of our models. It should be noted that due to the imbalanced nature of the ground truth Stratified K-folding, a method of guaranteeing that the ratio of positive to negative splits remained consistent across all splits, was necessary. This was done for every algorithm and every splitting methodology.

Finally, many of our algorithms were ensemble techniques that combined many base models to produce a more optimized predictive model. Both Bagging, a method by which multiple subsamples of the training algorithm are formed then aggregated and ranked into the most effective model, and Boosting, where after a round of 'boosting' misclassified predictors are reweighted to compensate for the error, ensemble methods were used. Ensemble methods come with possible drawbacks like overfitting, but after evaluation and multiple tuning efforts these methods proved efficacious

4.2 Clustering Methods

Clustering was performed using the following clustering algorithms; K-Means++, Agglomerative Clustering, DBSCAN, OPTICS and Spectral Clustering. We will briefly go over each algorithm for clarification. It should also be noted that clustering was performed using all eight features in calculating distance measures.

K-Means++: A partition-based method of clustering using a distance function and provided with a predicted number of n "clusters" (for us based on the number of expected classes), calculates the distance of each instance from a set of centroids, in our case randomly assigned, and then groups the instances based on whichever of these centroids they are closest to. This is repeated until several rounds has completed or no change is observed in the centroids for a certain number of rounds. K-Means is sensitive to outliers, has difficulty with concave data, and clusters of differing size when inter cluster distance is low.

Agglomerative Clustering: Is a form of hierarchical clustering, in this case a bottom-up approach where each instance starts in its own cluster and pairs of clusters are merged as one moves up the hierarchy. Linkage criteria is used to determine the distance between sets of instances, in our case 'Ward' linkage, which utilizes the increase in variance for the clusters being merged. As with K-Means a predicted number of clusters was provided to the algorithm

to match the number of expected classes. Agglomerative clustering uses a greedy approach and as a drawback can be computationally expensive.

DBSCAN and OPTICS: DBSCAN and OPTICS can be described together since OPTICS often uses the DBSCAN algorithm to perform its final clustering. We used both to verify if there would be any difference between the algorithms with the same hyperparameters. OPTICS is an algorithm that finds density-based clustering in data, it does not provide clusters, but a clustering order based on reachability and core distance for each instance in a data set. Cluster ordering and outliers/noise can be identified from the peaks and valleys of an associated plot of these distances. OPTICS was designed to compensate for the drawbacks in DBSCAN's ability to deal with varying density data. DBSCAN like OPTICS groups instances based on how closely packed they are into their neighborhoods. Both OPTICS and DBSCAN were selected by us to ascertain if the Pulsars might be mostly dissimilar and would appear as noise or outliers.

Spectral Clustering: Is a set of clustering techniques that use the "spectrum" of eigenvalues to form a similarity matrix of the data before performing dimensionality reduction followed by the creation of a completely connected graph. Clusters are linked based on the weight of their edges. Spectral clustering was picked as a data exploration technique because it is a connectivity technique and could be used on non-spherical data (centroid focused or clustering around a central point) and non-convex clusters. The criterion used by us was Nearest Neighbors and was the only algorithm that benefited from quantile normalization of the data set.

4.3 Clustering Results

The results of our clustering attempts are shown here. All methods of clustering showed promise for exploring our hypothesis that the features of the data set would prove to be as effective as the authors had claimed. From the table below you can see that K-Means, Agglomerative, and Spectral Clustering gave us the most accurate clustering results when compared with the ground truth. Fowlkes-Mallow and Rand Index were used to verify this:

Table 9. Clustering evaluation scores.

		Clustering Metrics		
	Clustering Algorithm	**Normalization Method**	**Rand Index**	**Fowlkes-Mallow**
0	K-means	z-score	0.934777	0.962299
1		robust scalar	0.849772	0.910860
2	OPTICS	z-score	0.880615	0.928114
3		robust scalar	0.833149	0.895164
4	Spectral Clustering	z-score	0.876046	0.924212
5		robust scalar	0.858046	0.913716
6		quantile scalar	0.943259	0.966618
7	DBSCAN Clustering	z-score	0.869196	0.920768
8		robust scalar	0.817732	0.884864
9	Agglomerative Clustering	z-score	0.910733	0.949834
10		robust scalar	0.855120	0.914222

To give a more intuitive look into the results of our clustering PCA (See section 3.3) was used for dimensionality reduction on our data then the clustering predictions were mapped to each instance. Visualizations were then created and can be seen for the three most successful clustering techniques in the following figures. It should be noted that the densest cluster in every visualization was composed of the non-pulsar class.

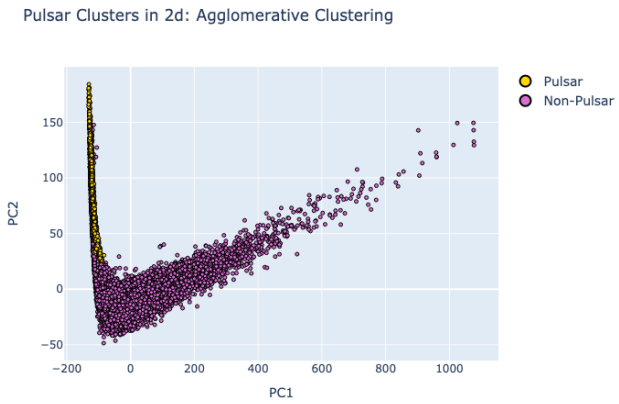

Figure 9. 2D visualization of agglomerative clustering

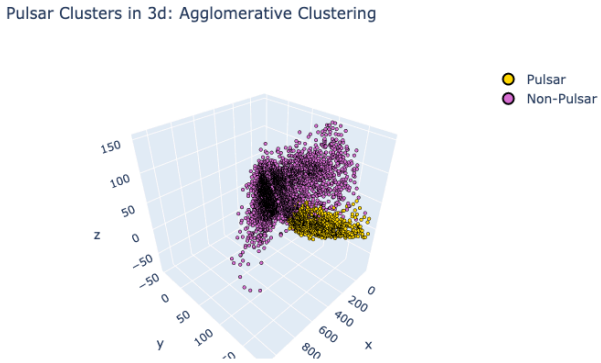

Figure 10. 3D visualization of agglomerative clustering

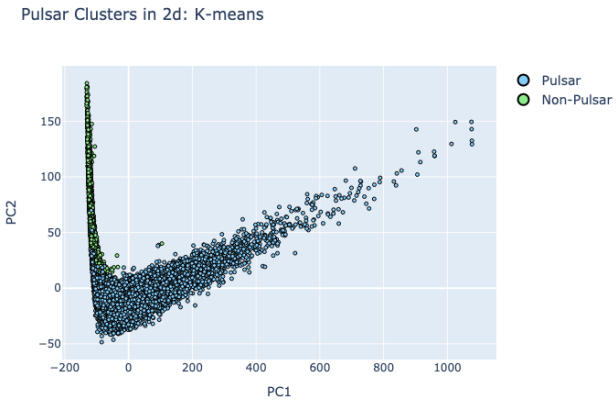

Figure 11. 2D visualization of K-Means clustering

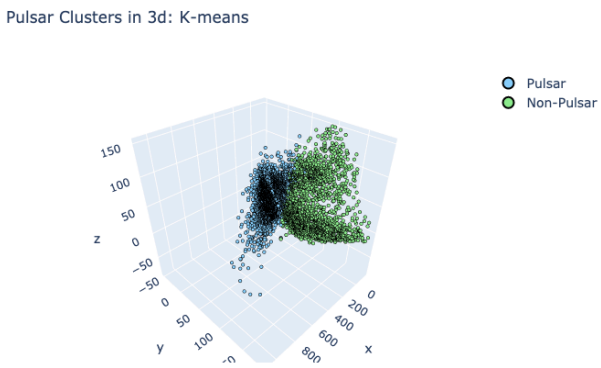

Figure 12. 3D visualization of K-Means clustering

Figure 13. 2D visualization of spectral clustering (highest ranked of the clustering methods tested)

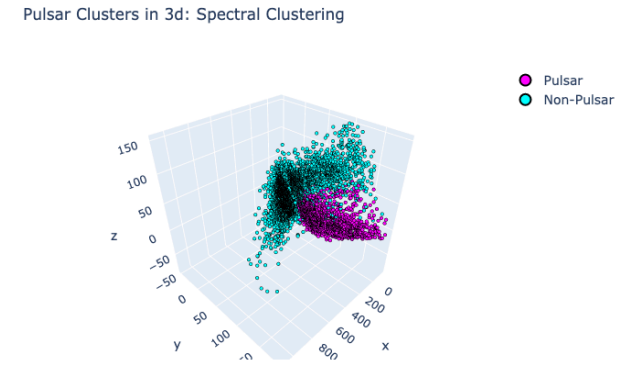

Figure 14. 3D visualization of spectral clustering (highest ranked of clustering methods tested)

Interactive versions of the 3D clustering can be found in the HTML files folder of the code repository associated with this project.

4.4 Classification Methods

Classification was performed on the data using Naive Bayes (also used by the paper), Random Forest, AdaBoost, and XGBoost. We will briefly go over each algorithm for clarification. Only Naive Bayes was used directly in the paper, but the other methods were either not available to us or are components of the ensemble methods we used in our research. During Cross Validation and Grid Search special care was taken to account for the imbalanced nature of the set and positive class weight offsets were used where available.

Naive Bayes Classification: A probabilistic classifier that uses Bayes' theorem and a "naive" assumption about the

independence of each predictor variable in a feature set. Classifying whether a tuple belongs to a binary class A or B, requires Naïve Bayes to look at each feature separately and then determine the proportion of previous tuples that belong to a class that have the same value as the current tuple for the feature being looked at. The process repeats for all the other features in the tuple and the classifier takes the product. This product is multiplied by the proportion of the class in the data and then checked as to whether this final value is greater than the corresponding calculation on the tuple for the alternative classes. Depending on whatever calculation is greatest determines to which class an instance belongs.

Random Forest: is an ensemble classifier that generates a set of decision trees during training and then outputs the predicted class that is the mode of the individual trees. Random Forest is a Bagging ensemble method and Random Forests are useful in that they can compensate for overfitting during training.

AdaBoost: Shorthand for Adaptive Boosting, Adaboost is a boosting ensemble classifier and is adaptive because each set of subsequent misclassified training instances are reweighted in favor of those instances misclassified by previous classifiers. Adaboost is often less susceptible to overfitting the training data. Though the individual learners are weak if the effectiveness of each learner is better than random guessing, the final model can become a strong learner.

XGBoost: Standing for Extreme Gradient Boosting, XGBoost is another ensemble classifier that uses both boosting and the benefits of decision tree based random forests. It differs from other boosting methods due to its special penalty metrics for trees, weight measures for handling imbalanced data sets giving increased/decreased weight to positive classes, extra randomization functions, and the ability to control tree height or implementing shrinkage of leaf nodes.

4.5 Classification Results

Here are the results of our classification evaluation. Scoring metrics have been defined earlier in the document (See section 4). We have also provided the best hyper parameters for some of the methods and splits used during classification (this is not an exhaustive list and is provided here only to illustrate the results of running grid search for each training, full hyperparameter listings for the best estimators can be found in the code repository associated with this report). **Note the increased number of hyperparameters associated with XGBoost**. The below tables use splits to compensate for the lack of additional training sets used by the paper, but only in attempting to improve upon the results from the same data set, not in reaching conclusions about the assumptions put forth in the original paper. All splits except the single experimental set used all the classification methods described in the previous section. We have provided the outcome of XGBoost model's training on this experimental set, but for the purposes of validating the work of the authors it will not be considered in the conclusions or discussions to follow.

Table 10. Final Scoring for all models across all splits. Darker colors indicate better scoring except FPR where lowest score is best.

Our Results

	Split	Algorithm	G-Mean	F-Score	Recall	Precision	Specificity	FPR	Accuracy	AUC
0	Lyons	XGBoost	0.862534	0.862526	0.858930	0.866153	0.988106	0.011894	0.977482	0.923518
1		Random Forest	0.709517	0.684211	0.930507	0.541010	0.541010	0.458990	0.929359	-
2		NB	0.897844	0.638882	0.873523	0.503606	0.922842	0.077158	0.918786	0.898183
3		AdaBoost	0.939231	0.780041	0.917999	0.678131	0.960954	0.039046	0.957421	0.939476
4	Group	XGBoost	0.842414	0.842349	0.852895	0.832061	0.992791	0.007209	0.987167	0.922843
5	5 Fold CV	XGBoost	0.876337	0.876336	0.875000	0.877676	0.987700	0.012300	0.977374	0.931350
6		Random Forest	0.882942	0.882629	0.859756	0.906752	0.906752	0.093248	0.979050	-
7		NB	0.907040	0.733850	0.865854	0.636771	0.950185	0.049815	0.942458	0.908019
8		AdaBoost	0.913639	0.876190	0.841463	0.913907	0.992005	0.007995	0.978212	0.916734
9	3 Fold CV	XGBoost	0.885600	0.885581	0.879852	0.891386	0.989191	0.010809	0.979177	0.934522
10		Random Forest	0.886998	0.886559	0.859519	0.915354	0.915354	0.084646	0.979854	-
11		NB	0.910090	0.739984	0.870610	0.643443	0.951360	0.048640	0.943965	0.910985
12		AdaBoost	0.911816	0.877907	0.837338	0.922607	0.992918	0.007082	0.978669	0.915128
13	10 Fold CV	XGBoost	0.869716	0.869565	0.853659	0.886076	0.988930	0.011070	0.976536	0.921294
14		Random Forest	0.872482	0.872274	0.853659	0.891720	0.891720	0.108280	0.977095	-
15		NB	0.900629	0.727273	0.853659	0.633484	0.950185	0.049815	0.941341	0.901922
16		AdaBoost	0.902813	0.857143	0.823171	0.894040	0.990160	0.009840	0.974860	0.906665

Table 11. Best hyperparameters from AdaBoost grid search and 3-fold CV

Best Parameters from AdaBoost 3 Fold CV

	Hyperparameter
learning_rate	0.200000
n_estimators	200.000000

Table 12. Best hyperparameters from Random Forest grid search and 5-fold CV

Best Parameters from Random Forest 5 Fold CV

	Hyperparameter
max_depth	15
min_samples_leaf	1
min_samples_split	15
n_estimators	200

Table 13. Best hyperparameters from XGBoost grid search and 3-fold CV

Best Parameters from XGBoost 3 Fold CV

	Hyperparameter
colsample_bytree	1.000000
gamma	1.500000
max_depth	3.000000
min_child_weight	10.000000
scale_pos_weight	2.000000
subsample	1.000000

For a more intuitive understanding of some of our results, included here are the confusion matrices for the best classifier using the Paper methodology and our best Classifier using 3-fold CV, 5-fold CV, and 10-fold CV:

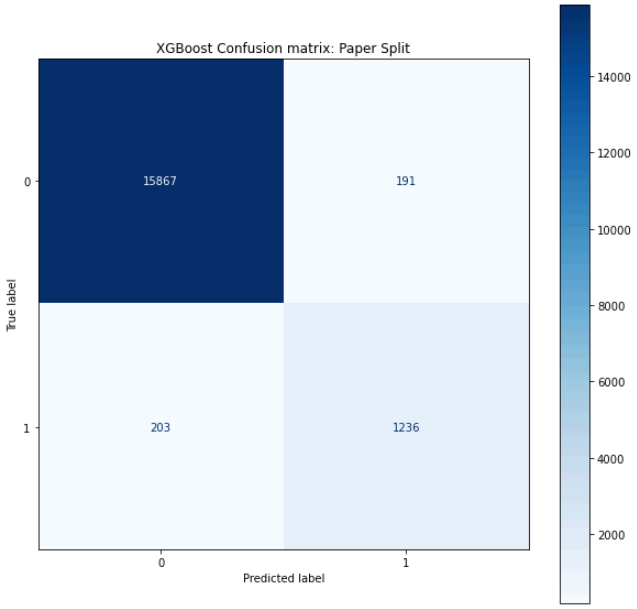

Figure 15. XGBoost confusion matrix. Best for paper training methodology.

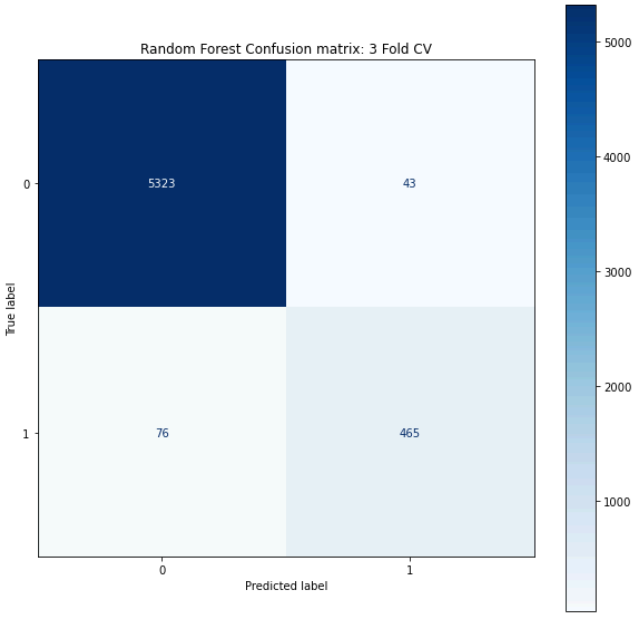

Figure 16. Random Forest confusion matrix. Best for 3-fold CV

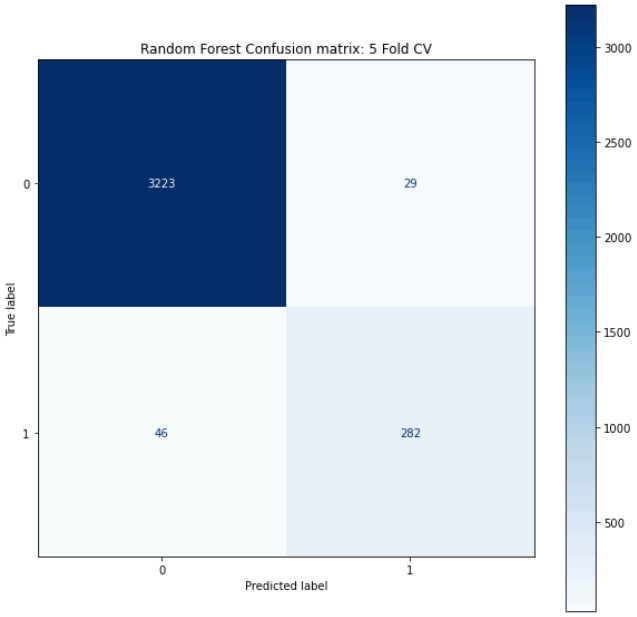

Figure 17. Random Forest confusion matrix. Best for 5-fold CV

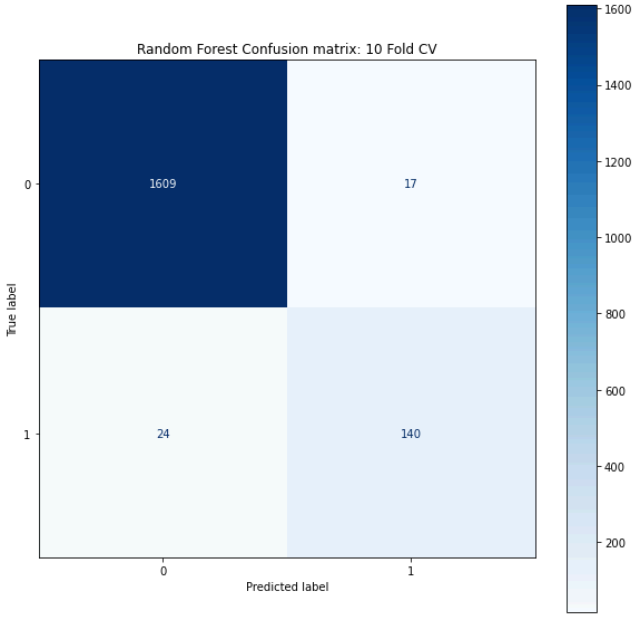

Figure 18. Random Forest confusion matrix. Best for 10-fold CV.

4.6 Feature Importance

One of the key points of the paper [3] was the value of specific features for identifying pulsars from the set of eight created by the authors. To verify their rankings, we performed feature importance testing using multiple methods outlined earlier in this paper (See section 3.4) one of those techniques was SHAP.

SHAP (SHapely Additive exPlanations) can be calculated for any tree-based model and is based on game theory techniques, namely the Shapley value of fair distribution of both gains and costs. Shap produces easy to understand feature rankings, but it should be noted that Shap values are not causal values but model interpretability values. Below is one of our Shap plots for feature importance testing.

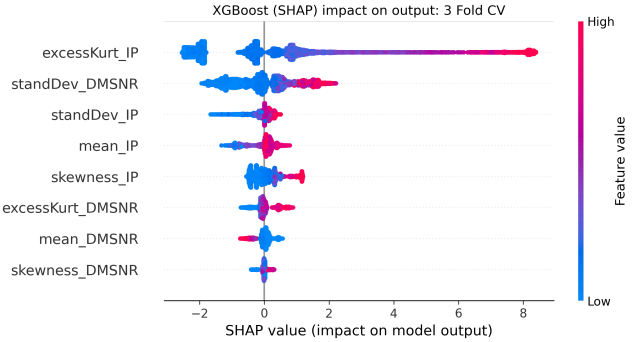

Figure 19. Shap output bee swarm plot of feature impact. Features are listed in descending order of impact and therefore importance.

We have created multiple plots using the feature importance identification techniques outlined above and from those created a basic table of the features in ranked order starting with the paper's conclusion, our violin plot interpretation, the results of Random Forest's built-in feature importance tools, and finally the results of Shap importance derived from XGBoost. Below is the plot from Random Forest and the final table ranking.

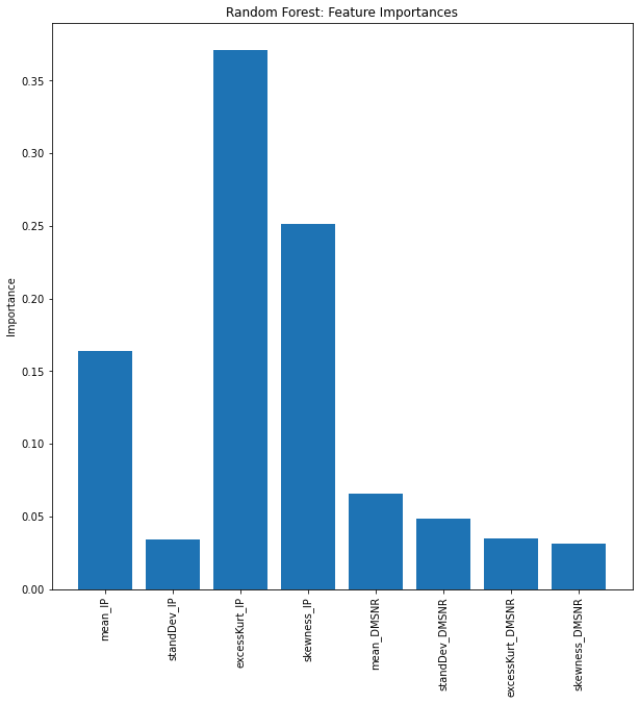

Figure 20. Random Forest built-in feature importance. Higher y-values indicate greater importance.

Table 14. Final observed feature importance ranking. Note similarities between Paper Ranking and XGBoost Ranking.

Observed Feature Importance Ranking

	Feature	Paper Ranking	Our Initial Ranking	Random Forest Ranking	XGBoost Ranking
0	excessKurt_IP	1	6	1	1
1	standDev_DMSNR	2	5	5	2
2	mean_IP	3	1	3	4
3	skewness_IP	4	8	2	3
4	standDev_IP	5	2	7	5
5	excessKurt_DMSNR	6	3	6	6
6	mean_DMSNR	7	7	4	7
7	skewness_DMSNR	8	4	8	8

5 Discussions and Conclusions

It is important to recall that the goal of our project is to confirm that the features chosen in the paper [3] prove to be important, as well as provide additional classification methods in hopes to gather better classification results.

5.1 Feature Conclusions

Refer to Table 14 to see our observed feature importance ranking from our feature importance testing described in sections 3 and 4. Notice that the top four features in the paper [3] were also the top four features in our XGBoost Classification feature ranking. The Random Forest ranking was also close on matching the most important features of the paper [3]. From our results, we can conclude that the features chosen by Lyon [3] prove that their feature selection method described in section 2.1 is an improved way of determining pulsars from other feature selection methods.

5.2 Classification Conclusions

Following the discussion in section 4.6, our results from Table 10 indicate that we were able to achieve high classification performance using the feature selection and new classification algorithms on the HTRU2 dataset. We see this conclusion based on how our classification algorithms generally exhibited high levels of specificity and low false positive rates. The exception being the Random Forest Classifier for the Lyon split. What is also important to observe is that our classification results are consistent across all three folds. In pursuit of our stated goals, the metric that we placed more emphasis on was F-Score. This metric is described in detail in section 4.1. Referencing back to Table 5 and Table 14 to compare and evaluate F-Score results, we notice that our classification models produce a higher accuracy in comparison to F-Score results from the paper [3]. To further evaluate F-Score metrics for our three folds, we determined that the Random Forest Classifier consistently outperformed among all classification algorithms. The more generative Naive Bayes classifier generally underperformed among all classification algorithms. For an overall evaluation of classification algorithms, we noticed that the AdaBoost classifier outperformed in all metric results. In conclusion, the goal to evaluate different classification algorithms and provide more accurate classification results was achieved.

5.3 Discussion on Limitations

In the paper [3], multilayer perceptron (MLP) and support vector machine (SVM) algorithms were used as classification techniques. MLP is a feedforward artificial neural network and is characterized by several layers of input nodes connected as a direct graph between the input and output layers. Because the methodology was not explained in the paper and we did not know how many hidden layers were present, we did not consider the classification technique. Another reason is due to MLP and SVM (defined in the next section) being beyond the scope of evaluation within this course.

6 Possible Future Work and Lessons Learned

Support Vector Machine (SVM) is a supervised machine learning algorithm used primarily for classification but can also be used for regression. SVM produces good accuracy with smaller amounts of computing power compared to other machine learning methods, making it a good starting point for further work.

Another path forward would be to train our models with more datasets. If any more data was made public, that data could be incorporated and used to potentially refine the models.

Feature importance was heavily emphasized in all aspects of this research. More exploration beyond our simple testing using true statistical analysis might be an ideal candidate for future work on this data set. Questions that could be answered: Based on the observed high correlation between some features, could the set be reduced to fewer features? Since spectral clustering was our best clustering performer and since it inherently contains dimensionality reduction, could this also indicate that features could be removed or combined?

ACKNOWLEDGMENTS

Special thanks would like to be given to Professor Chengcui Zhang and the TA Nipurn Jain for all their assistance and guidance for the last few months. Additional thanks would like to be given to the authors and researchers whose information was cited in this report.

REFERENCES

[1] Cofield, Calla. 2016. What Are Pulsars? (April 2016). Retrieved April 20, 2021 from https://www.space.com/32661-pulsars.html

[2] "Facts and figures". SKA Organization. Archived from the original on 28 July 2012. Retrieved 26 May 2012.

[3] R. J. Lyon, B. W. Stappers, S. Cooper, J. M. Brooke, J. D. Knowles, Fifty Years of Pulsar Candidate Selection: From simple filters to a new principled real-time classification approach, Monthly Notices of the Royal Astronomical Society 459 (1), 1104-1123, DOI: 10.1093/mnras/stw656

[4] R. J. Lyon (2016): HTRU2. figshare. Dataset. https://doi.org/10.6084/m9.figshare.3080389.v1

[5] R.J. Lyon, Brooke J. M., Knowles J. D., Stappers B. W., 2014, 22nd Int. Conf. on Pattern Recognition, p. 1969

[6] minc33. 2018. Visualizing High Dimensional Clusters. (October 2018). Retrieved April 20, 2021 from https://www.kaggle.com/minc33/visualizing-high-dimensional-clusters

[7] A. Scaife, 2019. Pulsar Classification Using a Random Forest. Cern School of Computing 2019. Retrieved April 19, 2021 from https://as595.github.io/classification/.