Identifying the Existence of Pulsar Stars

Group 6 Wang, Xinwei xiw180@ucsd.edu Yu, Yue yuy079@ucsd.edu Zheng, Zihan zizheng@ucsd.edu

ABSTRACT

A pulsar is a highly magnetized rotating neutron star that emits a beam of electromagnetic radiation detectable on Earth. Astronomers and scientists use pulsars throughout our galaxy as a giant scientific instrument to directly detect gravitational waves and a very precise timepiece to keep time accurately. Therefore, being able to identify the existence of pulsar stars with high accuracy is intriguing. In our project, we pre-processed the University of Manchester's HTRU2 dataset [1] by balancing the data using a statistical approach which will be explained later. We tried various models, including logistic regression classifier, random, forest classifier, convolutional neural network, and multi-layer perceptron classifier to predict pulsar stars, and we got high accuracy, precision, and recall from all 4 models.

1. INTRODUCTION

Pulsars are a special type of Neutron star that can produce radio emission. They are of significant interest by scientist because they are used as an instrument to detect gravitational waves and to accurately keep space-time. When a pulsar rotates, it produces a detectable pattern of radio emission, which is very precise and repeats periodically [2]. Therefore, we can use the pattern of radio emission signal to decide whether a pulsar exists at a given location. However, in practice, radio frequency interference and noise can produce signals that are similar to that of pulsars, so it's highly intriguing to come up with a way to classify between pulsars and radio frequency interference or noise.

In this paper, we are going to utilize four machine learning models to automatically detect the existence of pulsar stars. From the HTRU2 dataset, the input to our algorithm is an eight-dimension vector, which is pre-processed by UCI from some raw signal data. The feature set contains mean, standard deviation, excess kurtosis, and skewness of the integrated profile and mean, standard deviation, excess kurtosis, and skewness of the DM-SNR curve. The following Dataset section in 3.1 will explain these physic properties in detail.

We then use mainly 4 different models, including logistic regression classifier, random, forest classifier, convolutional neural network, and multi-layer perceptron classifier to output a predicted label, which is either 1 or 0 (1 stands for pulsar and 0 stands for non-pulsar). Finally, we measure the performance of the models based on the accuracy, precision, and recall for both positive and negative labels.

In addition to the build of models, we also pre-process the HTRU2 dataset by adding more data with positive labels (pulsars) because the positive and negative labels in the HTRU2 dataset are unbalanced, which will also be explained in 3.1. Without this step, our models can still get a good accuracy by predicting 0 for all samples, but this will lead to a pretty bad precision score.

2. RELATED WORK

One of the difficulties encountered during the searches for pulsars in the past fifty years is the analysis needed for the increasing number of 'candidate' pulsar detections arising from an increasing volume of data to be searched [3]. Our study focuses on solving this difficulty using various machine learning techniques to maximize the accuracy, precision, and recall for both positive and negative labels on both the original dataset and the generated fake dataset. There are several prior works that have investigated this problem. In this section, we discuss the prior work, comparing and contrasting where appropriate.

2.1 New Candidate Features Selection

In Fifty years of pulsar candidate selection: from simple filters to a new principled real-time classification approach [4], Lyon et al. use statistical classification to extract 8 new features for the purpose of minimizing the number of features without reducing classification performance. These features will be described in Section 3.2. Lyon et al. then use Gaussian Hellinger Very Fast Decision Tree (GH-VFDT) to classify multiple pulsar star datasets including HTRU2. Their results are as below:

Table 1: Partial Result Table from Lyon 2016[4]

Algorithm	Accuracy	Precision	Recall	Specificity
GH-VFDT	0.978	0.899	0.829	0.992

Our study builds on classifying based on this 8 features that Lyon et al. extracts, while we use more sophisticated machine learning models to achieve higher accuracy, precision, recall, specificity (fraction of negatives correctly identified), and also negative precision (Fraction of retrieved negative instances that are positive) than Lyon et al. do with decision tree.

2.2 Past Methods & Models

There have been several attempts to solve this pulsar star candidate selection problem. Eatough et al. and Bates et al. use Artificial Neural Networks (ANNs) to approach the problem [5][6]. Their models are effective to some extent but

are not as sophisticated as the neural networks we use nowadays like Multi-Layer Perceptron (MLP) and Convolutional Neural Networks (CNN). PEACE uses a set of algorithms to analyze a pulsar candidate and calculate a score, which is a measure of how likely a candidate is to be a real pulsar [7]. Compared to other studies, PEACE is fully pre-determined by six quality factors [7] and focuses on significantly increasing the rate of identifying pulsars rather than increasing the prediction accuracy, precision, and recall. SPINN, an automated pulsar candidate classifier designed with maximum recall in mind, is another ANN that produces a real-valued and continuous output instead of a binary class label, which can also produce a subjective ranking of candidates that can be used to prioritize visual inspection [8]. Our study has a different goal than that of SPINN: we want to use modern machine learning models to maximize not only recall but also positive/negative precision and specificity.

3. DATASET AND FEATURES

3.1 Dataset

HTRU2 is a dataset which describes a sample of pulsar candidates collected during the High Time Resolution Universe Survey (South) [1] which describes pulsar candidates collected during the High Time Resolution Universe Survey. Each candidate in the dataset contains 8 attributes: Mean, Standard deviation, Excess kurtosis, and Skewness of the integrated profile, and Mean, Standard deviation, Excess kurtosis, and Skewness of the DM-SNR curve. Candidates are labeled with the ground truth of whether a pulsar star exists or not.

There are 17898 total examples in the dataset. 16239 of them represent signals caused by noise or radio frequency interference, which is roughly 90%. 1639 of the data represent pulsars, which is only 10%. There is a necessity to balance the dataset so that we have roughly 50% data with positive labels and 50% data with negative labels or our models may overfit the data by predicting accurately for negative labels but poorly for positive labels.

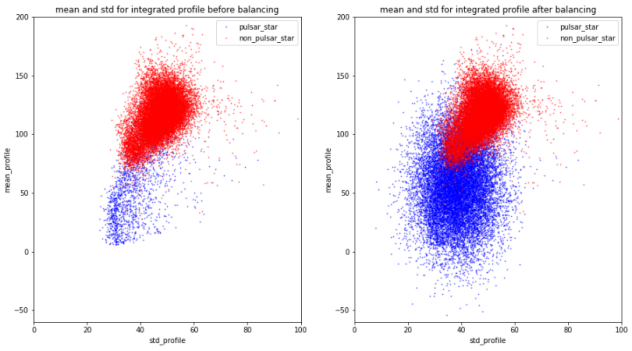

Figure 1: before and after balancing the dataset

To balance our data, we need to generate more data with positive labels, so we gather all data with positive labels and calculate the mean and standard deviation of all eight attributes. Then we randomized each attribute with the given attributes' normal distribution generated using their mean and standard deviation and find the 9 nearest neighbors of the generated data if more than half of the neighbors have positive labels, we keep the data as one row of data. In this way, we finally generate enough data to make a new dataset with 50% positive labels and 50% negative labels. As figure 1 shows, the newly generated dataset looks balanced and convincing.

3.2 Features

We have 8 features and they are all directly derived from integrated profile and DM-SNR (signal-to-noise ratio) curve. They are radio emission-related physics property and thus are appropriate for predicting the existence of pulsar stars. The integrated profile is an array of continuous variables that describe a longitude-resolved version of the signal that has been averaged in both time and frequency [4]. SNR, the signal-to-noise ratio, measures the ratio between the level of signal and level of noise, and DM-SNR curve is a combination between the real and theoretical acceleration-SNR curve [9]. The description and definition of the 8 features are in Table 2:

Table 2: The eight features derived from the integrated pulse profile P and DM-SNR curve D from Lyon 2016[4]

Feature	Description	Definition
$Prof.\mu$	Mean of P.	$\frac{1}{n}\sum_{i=1}^{n} p_i$
$Prof.\sigma$	Standard deviation of P.	$\sqrt{\frac{\sum_{i=1}^{n}(p_i-\bar{P})^2}{n-1}}$
$Prof.k$	Excess kurtosis of 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 P.	$\frac{\frac{1}{n}(\sum_{i=1}^{n} p_i-\bar{P})^3}{(\sqrt{\frac{1}{n}(\sum_{i=1}^{n}(p_i-\bar{P})^2)})^3}$
$DM.\mu$	Mean of D.	$\frac{1}{n}\sum_{i=1}^{n} d_i$
$DM.\sigma$	Standard deviation of D.	$\sqrt{\frac{\sum_{i=1}^{n}(d_i-\bar{D})^2}{n-1}}$
$DM.k$	Excess kurtosis of 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 D.	$\frac{\frac{1}{n}(\sum_{i=1}^{n} d_i-\bar{D})^3}{(\sqrt{\frac{1}{n}(\sum_{i=1}^{n}(d_i-\bar{D})^2)})^3}$

Below are 2 examples from the dataset. One is not a pulsar, and the other represents a pulsar.

140.562500	55.683782	-0.234571	-0.699648	3.199833	19.110426	7.975532	74.242225

Figure 2: feature vector with negative label

99.367188	41.572202	1.547197	4.154106	27.555184	61.719016	2.208808	3.662680

Figure 3: feature vector with positive label

4. METHODS

4.1 Logistic Regression

The logistic regression model uses a logistic function to model the output y. We train a logistic regression model with binary class L2 regularization weight because the data is

not sparse. Using this model, we minimize the loss function $L(w,b)$, which formula is shown below:

$$L(w,b) = \sum_{i=1}^{n} \ln(1 + e^{-y^{(i)}(w \cdot x^{(i)} + b)})$$

Then we use the gradient descent to update the weight to minimize the loss function with the following formula:

$$w_{t+1} = w_t + \eta \sum_{i=1}^{n} y^{(i)} x^{(i)} Pr_{w_t}(y^{(j)} | x^{(i)})$$

where η is the step size.

After iterations, eventually, we have a better weight for our logistic regression model.

4.2 Random Forest

Random Forest is a method using multiple decision trees to train the model and make the output be the mean prediction result of all decision trees generated. It is better than decision tree classifier because it can better prevent data over-fitting.

For each iteration, n points would be chosen randomly, with replacement from the data set. Then they are fitted into a decision tree, where at each node restrict to one of $\sqrt{d}$ features chosen at random, where d is the dimension of the data.

The final predictor would be the majority vote of those iterations' decision trees.

4.3 Multi-layer Perceptron

A Multi-layer Perceptron (MLP) is a logistic regression classifier. A perceptron classifies its input x, which is a feature vector, by separating 2 classes with a line. It produces its output y by the following formula:

$$y = \varphi(\sum_{i=1}^{n} w_i x_i + b) = \varphi(w^T x + b)$$

In the formula above, w is the weight vector, b is the bias and φ is the activation function. An MLP is composed of more than one perceptron. The input layer takes in the signal, and the output layer makes a decision about what class the input belongs to.

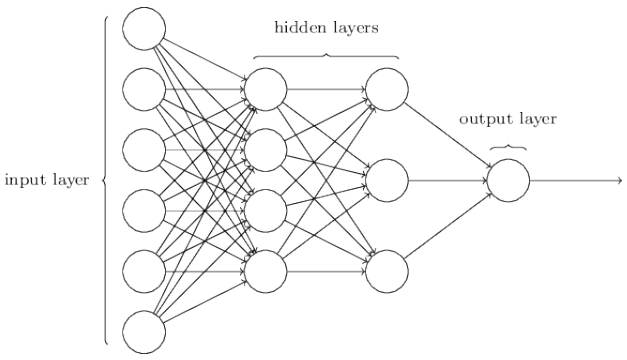

Figure 4: An MLP network example

4.4 Convolutional Neural Network

A Convolutional Neural Network (CNN) is a regularized version of multi-layer perceptrons. The network has many different types of layers that serve variable roles. A Convolutional layer takes a tensor with shape as input and outputs a convolved feature whose dimensionality either increases, remains the same, or decreases. Following the convolutional layer, an activation function is usually applied to the output tensor. For example, the figure below demonstrates how ReLU, one kind of activation function works. ReLU takes in a tensor and convert all negative values to zero and keep all positive values.

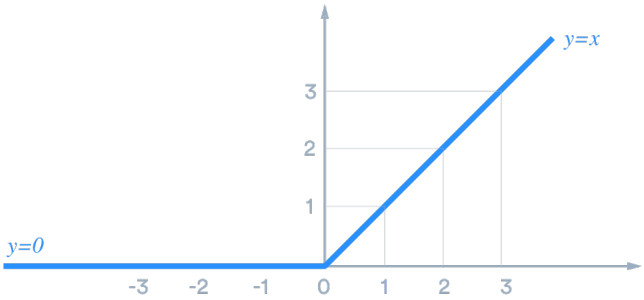

Figure 5: ReLU function

A pooling layer reduces the spatial size of a convolved feature, and it's usually added after a convolutional layer. A fully connected layer is usually a method to learn low-level features and classify them with Softmax Classification, which outputs a categorical probability distribution. The following formula shows how a softmax function squashes each output unit of the fully connected layer to a value between 0 and 1:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}}$$

Our CNN model uses Stochastic gradient descent (SGD) algorithm to minimize the training loss. In each iteration, the weight matrix is updated according to the following formula:

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta; x^{(i)}; y^{(i)})$$

In the above formula, η is the step size and J is the loss function. For J, the model uses binary cross-entropy to measure loss over iterations, which the formula is:

$$-(y \log(p) + (1-y) \log(1-p))$$

where p is the prediction and y is either 1 or 0.

5. RESULTS

5.1 Experiments

For all models, we first trained with the original HTRU2 dataset. Then we applied the statistical random data sampling method mentioned in section 3.1 to create more data with positive labels, which means we added more "pulsars" and trained with the new dataset.

For the logistic regression, in order to better tuning the parameters, we use grid search with cross-validation and nfold = 5, to find the best C value from C_list where $C\_list = [0.001, 0.01, 0.1, 1, 10]$. After the grid search, we found that

	Accuracy	Negative Precision	Positive Precision	Negative Recall	Positive Recall
LR	0.972975	0.9656244	0.9806157	0.9810514	0.9648466
RF	0.988444	0.9847004	0.9922729	0.9923834	0.9844801
CNN	0.983692	0.9751824	0.9925728	0.9927549	0.9745699
MLP	0.985742	0.9768417	0.9950447	0.9951699	0.9762528

Table 3: Performance metrics for the 4 models

the best C is 10. So we used C = 10 for the logistic regression model. For the random forest, we followed the same way of doing grid search with nfold = 5, to find the best parameters listed below:

We chose best n_estimators 's value from n_estimator_list where $n\_estimator\_list = [10, 30, 50, 100]$.

We chose best max_depth 's value from max_depth_list where $max\_depth\_list = [3, 5, 7, 9, 11]$. After the grid search, we found the best n_estimators is 100 and the best max_depth is 11, thus we used those two parameters to train the random forest model.

For the multi-layer perceptron, we tried tanh and ReLU for activation function. We tried adam and stochastic gradient descent for weight optimization solver, and we tried different alpha, the L2 penalty for the regularization term, in our experiment. After the experiments, we found that the combination of adam solver, 0.1 alpha, and ReLU activation creates a model with the best performance, which will be discussed in the next section.

For the convolutional neural network, we first tried different number of convolutional layers from 1 to 4. Different filter sizes, from (6, 2, 1) to (32, 4, 1) were experimented. We also tried to add and to not add a max pooling layer to each of the convolutional layer. For the activation function, we tried tanh and ReLU. A Dropout layer was added after all convolutional layers to avoid overfitting. For weight optimization solver, we tried adam and stochastic gradient descent.

5.2 Performance

Our primary metrics are accuracy, precisions for both positive and negative labels, and recalls for both positive and negative labels. Detailed results are listed in table 1 below. Our random forest model predicts with the best accuracy of 98.84%, negative precision of 98.45%, and positive recall of 98.45%

For the CNN model, we also measured the training loss and accuracy over iterations/number of epochs, and they are plotted below in Figure 6 and 7. Basically, the accuracy goes up steadily and the training loss decreases as the model iterates, which is what we expected because our sgd method aims to minimize the loss in every step of training.

The confusion matrix describing the outcome of the binary classification is described in Figure 8 to Figure 11. We evaluate our model with several attributes described in Table 4. The actual performance the models achieves are listed in Table 3.

From the performance evaluations, we can see that all 4 models reach state-of-art performance for all 5 metrics we evaluate. All the performance attributes for all 4 models are above 97 percent.

Among all 4 models we use, Random Forest achieves the best negative precision, accuracy, and positive recall while

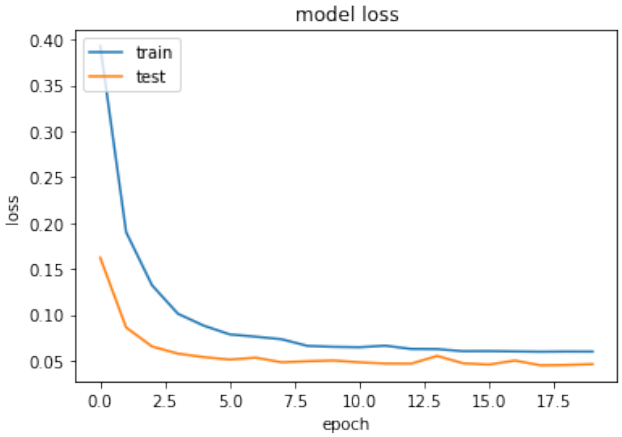

Figure 6: CNN training loss vs num epoch

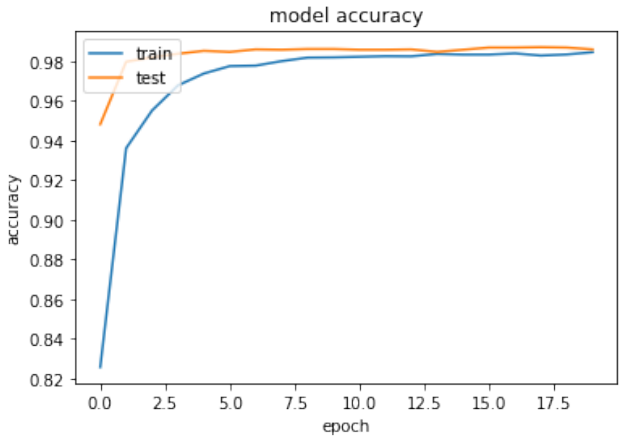

Figure 7: CNN accuracy vs num epoch

Multi-Layer Perceptron achieves the best negative recall and positive precision.

5.3 Discussion

Initially, our models' test accuracies were very high because the data was unbalanced (90 % with negative labels and 10 % with positive labels). As a result, we have relatively low precision and recall for Pulsars (positive label). After sampling more data with positive data, our models' performance dropped as expected. After fine tuning the models, we were able to achieve similar accuracy as what we had for the unbalanced data and higher precision and recall.

We found that excess kurtosis is the feature that contributes the most in identifying a Pulsar, but using just kurtosis for

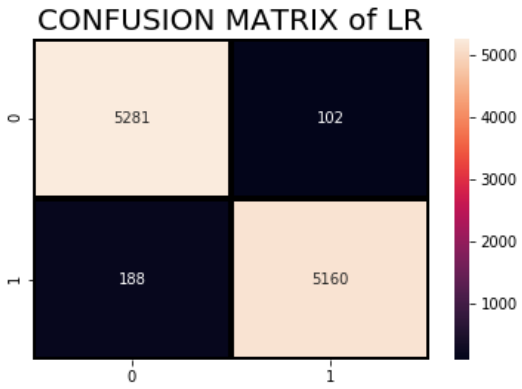

Figure 8: confusion matrix of logistic regression

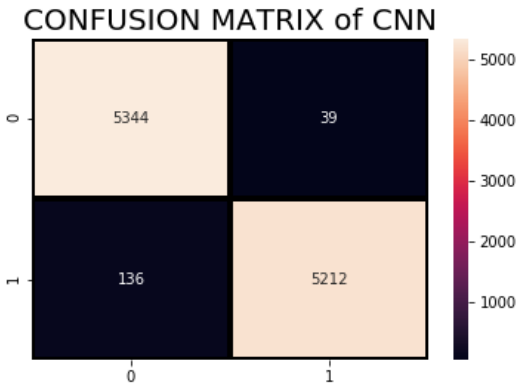

Figure 10: confusion matrix of random forest

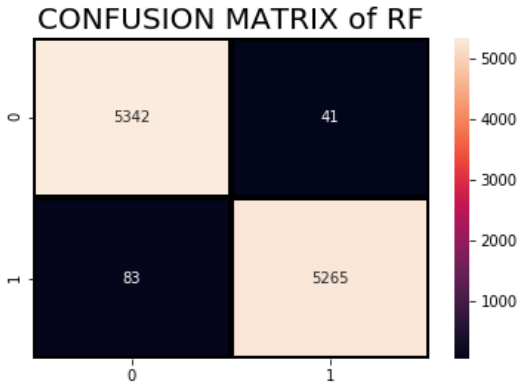

Figure 9: confusion matrix of random forest

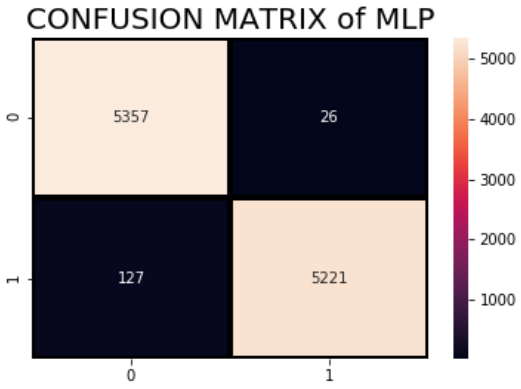

Figure 11: confusion matrix of random forest

integrated profile and DM-SNR curve does not improve the performance.

We expected Logistic regression and random forest to be capable of identifying a Pulsar because these two models have gained popularity for binary classification tasks. However, we did not expect CNN, which is well-known for image classification, to have an impressive performance.

6. CONCLUSION & FUTURE WORKS

6.1 Conclusion

In this study, we implement 4 machine learning models: Logistic Regression, Random Forest, Multi-layer Perception (MLP), and Convolutional Neural Network (CNN) to solve the pulsar star candidates identification problem. We use a statistical method to generate a balanced dataset to test the 4 models we implement. All 4 models achieve state-of-art performance for all 5 performance metrics we measure. Among the machine learning models, RF performs the best on the total accuracy of 98.84% while other models are only slightly worse than it after fine-tuning the parameters for these 4 models. Also, the random forest performs the best on the negative precision of 98.47% and positive recall of 98.45%. We believe that these models will all be useful, and especially random forest model, for future pulsar star identification in the area of astronomy.

6.2 Future Works

There are several future works that can be done if given more time for this study:

- Test our models in more pulsar star candidate datasets other than HTRU2, like HTRU1 and LOTAAS1 described in Lyon 2016 [4]
- Generate a better dataset, include more features that are in the original HTRU2 datasets and key representations of Pulsars and is larger and balanced. Use better balancing methods to generate better distributed fake data.
- Use our model predictions to discover actual Pulsar Stars in the universe like previous studies did [5][6]. If pulsar stars can be found based on our predictions, our machine learning models are proved to be useful in application.

7. CONTRIBUTIONS

Table 4: Evaluation Attributs describing the outcome of the classification

Statistic	Definition
Accuracy	$\frac{(TP+TN)}{(TP+FP+TN+FN)}$
Negative Precision	$\frac{TN}{(TN+FN)}$
Positive Precision	$\frac{TP}{(TP+FP)}$
Negative Recall (Specificity)	$\frac{TN}{(TN+FP)}$
Positive Recall	$\frac{TP}{(TP+FN)}$

7.1 Xinwei Wang

Created the logistic regression model. Investigated previous works for inspirations. Participated in parameter-tuning process. Worked on Related Works, Features, Results, Conclusion, and Future Works sections.

7.2 Yue Yu

Created the Random forest model. Implemented the algorithm to balance the data and drew the figure of showing how data looks. Implemented the grid search algorithms to help better tuning the parameters. Worked on Dataset and Features, methods and results for logistic regression and random forest sections.

7.3 Zihan Zheng

Created the MLP and CNN models. Worked on Introduction, Dataset and Features, methods and results for MLP and CNN, and Discussion sections.

8. REFERENCES/BIBLIOGRAPHY

[1] M. Keith, A. Jameson, W. Van Straten, M. Bailes, S. Johnston, M. Kramer, A. Possenti, S. Bates, N. Bhat, M. Burgay, *et al.*, "The high time resolution universe pulsar survey–i. system configuration and initial discoveries," *Monthly Notices of the Royal Astronomical Society*, vol. 409, no. 2, pp. 619–627, 2010.

[2] D. R. Lorimer, M. Kramer, *et al.*, *Handbook of pulsar astronomy*, vol. 4. Cambridge university press, 2005.

[3] R. J. Lyon, *Why are pulsars hard to find?* PhD thesis, The University of Manchester (United Kingdom), 2016.

[4] R. J. Lyon, B. Stappers, S. Cooper, J. Brooke, and J. 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*, vol. 459, no. 1, pp. 1104–1123, 2016.

[5] R. P. Eatough, N. Molkenthin, M. Kramer, A. Noutsos, M. Keith, B. Stappers, and A. Lyne, "Selection of radio pulsar candidates using artificial neural networks," *Monthly Notices of the Royal Astronomical Society*, vol. 407, no. 4, pp. 2443–2450, 2010.

[6] S. Bates, M. Bailes, B. Barsdell, N. Bhat, M. Burgay, S. Burke-Spolaor, D. Champion, P. Coster, N. D'Amico, A. Jameson, *et al.*, "The high time resolution universe pulsar surveyâĂŤvi. an artificial neural network and timing of 75 pulsars," *Monthly Notices of the Royal Astronomical Society*, vol. 427, no. 2, pp. 1052–1065, 2012.

[7] K. Lee, K. Stovall, F. Jenet, J. Martinez, L. Dartez, A. Mata, G. Lunsford, S. Cohen, C. Biwer, M. Rohr, *et al.*, "Peace: pulsar evaluation algorithm for candidate extraction–a software package for post-analysis processing of pulsar survey candidates," *Monthly Notices of the Royal Astronomical Society*, vol. 433, no. 1, pp. 688–694, 2013.

[8] V. Morello, E. Barr, M. Bailes, C. Flynn, E. Keane, and W. van Straten, "Spinn: a straightforward machine learning solution to the pulsar candidate selection problem," *Monthly Notices of the Royal Astronomical Society*, vol. 443, no. 2, pp. 1651–1662, 2014.

[9] R. P. Eatough, N. Molkenthin, M. Kramer, A. Noutsos, M. J. Keith, B. W. Stappers, and A. G. Lyne, "Selection of radio pulsar candidates using artificial neural networks," *Monthly Notices of the Royal Astronomical Society*, vol. 407, pp. 2443–2450, 07 2010.