Finding New Pulsars Using Machine Learning

Jacob Ian Matthews

Draft

May 25, 2020

Abstract

Contents

1 Introduction

1.1 Aim

This project consists of three aims:

i. Investigate the use of Machine Learning (ML) techniques in surveying pulsars;

ii. Create a training dataset for a Machine Learning algorithm to find pulsars in data obtained by the Murchison Widefield Array (MWA); and

iii. Evaluate the utility of the Machine Learning algorithm used by the LOFAR Telescope for the Murchison Widefield Array, and adjust the algorithm as necessary to achieve optimum pulsar candidate generation.

1.2 Structure of this Report

In this report, I will first explain in *Section 1.3* how Pulsars, Radio Astronomy, and Machine learning work, and then explain what has been dubbed the "Candidate Selection Problem" (**lyon**) and why Machine Learning is necessary in completing future pulsar surveys.

In *Section 2* I discuss the methods undertook in: (i) developing the machine learning training dataset for the Murchison Widefield Array algorithm, and (ii) evaluating the machine learning algorithm used by the LOFAR surveys for use with the Murchison Widefield Array.

In *Section 3* I analyse the results and findings obtained by the methods described in *Section 2*, and in *Section 4* I will discuss (i) the efficacy of the training datasets, and (ii) the usefulness of the LOFAR machine learning algorithm with the Murchison Widefield Array and why changes were made to the algorithm.

This report will end with my final conclusions on the use of machine learning in discovering new pulsars (*Section 5*), and my recommendations to future researchers undertaking a similar project (*Section 6*).

1.3 Background Theory

To answer the question of "what is a pulsar?" we must first investigate the evolution and death of stars.

A star can form when a cloud of hydrogen gas in the interstellar medium (ISM) collects mass over millions of years; as the mass of the gas cloud increases, its gravitational pull to gather more mass also increases (**maoz**). This protostar will eventually reach a critical mass in which the pressure of gravity within the gas causes enough friction between the gas particles to generate the required heat (thermal pressure) to begin fusing the hydrogen atoms into helium (**maoz**). This marks the beginning of the star's main sequence lifetime. Once the star has fused all of the available hydrogen gas in its core, the star will begin fusing helium into carbon and its outer envelope will expand, moving the star into

its "red-giant" phase (**maoz**). If the initial mass of the star was greater than 8 times the mass of the Sun (i.e. $8M_{\odot}$), the star will continue to fuse the elements in its core until it reaches a core of iron. At this point phenomena called nuclear photodisintegration and neutronization occurs, the latter of which causes electrons and photons to combine and form neutrons and anti-neutrinos (**maoz**). Neutronization can be shown as:

$$e^{-} + p \rightarrow n + \nu_e$$

This process removes the electron degeneracy pressure in the core of the star (a pressure which balances the star's gravitational pressure), causing the star to collapse under its own gravity in a timeframe of 0.1 seconds (**maoz**). The gravitational collapse stops once the gravitational pressure of the star is balanced by the neutron degeneracy pressure, i.e. the pressure from pushing neutrons together. The remaining star is incredibly dense, with a mass of approximately $1.4M_{\odot}$ and a radius of around 11km. This is called a neutron star (**maoz**).

Prior to the collapse of the star, we can imagine the star to be rotating at an angular velocity of ω_1. We know from the conservation of angular momentum that when the radius of a rotating object decreases, the angular velocity will increase (a spinning ice skater pulling their arms in close increases the speed of their spinning). We can therefore show that the angular velocity of the star after the gravitational collapse, ω_2, is much greater than the prior angular velocity:

$$L_1 = L_2$$

where L is the angular momentum, $L_1 = I_1\omega_1$ and $L_2 = I_2\omega_2$. Therefore:

$$I_1\omega_1 = I_2\omega_2$$

$$\omega_2 = \frac{I_1}{I_2}\omega_1$$

Assuming the star is a sphere, its moment of inertia, I is:

$$I = \frac{2}{5}MR^2$$

where M is the mass of the star and R is the radius of the star. We can thus show:

$$\omega_2 = \frac{\frac{2}{5}MR_1^2}{\frac{2}{5}MR_2^2}\omega_1$$

$$\omega_2 = \left(\frac{R_1}{R_2}\right)^2 \omega_1$$

where $R_1 \gg R_2$. We are left with a neutron star with a very large angular velocity. Analogous to the angular velocity of the star, the magnetic field of the star is also amplified. The ionised gas in the iron core of the star, which

generates a magnetic field, is compressed by the gravitational collapse, forcing the flux of the magnetic field to be amplified such that the field strength is approximately 10^{10} times stronger in the neutron star compared to during the star's main sequence lifetime (**maoz**).

If the rotation of the neutron star is misaligned with the axis of the magnetic field by an angle θ, the spinning magnetic dipole will radiate electromagnetic waves (**maoz**). As the neutron star rotates, the radiated electromagnetic waves will periodically sweep across the line of sight of an observer, creating a pulse of light. See Figure **??**. We can therefore define a pulsar as a rapidly rotating neutron star that appears to periodically emit electromagnetic waves (**maoz**; **lorimer**; **swainston**).

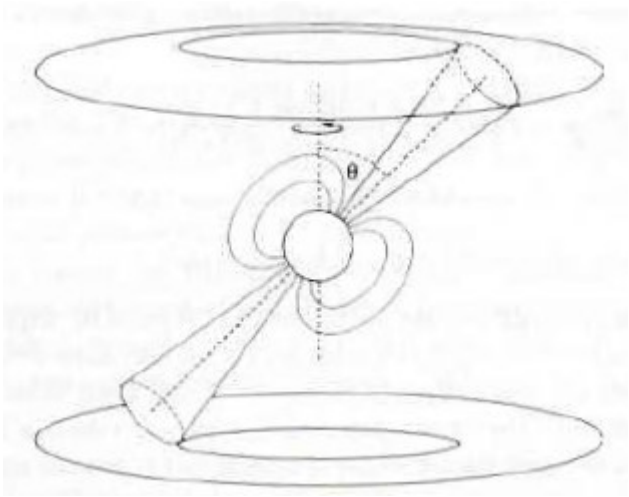

Figure 1: A pulsar (**maoz**).

While some pulsars, like the Crab Pulsar (**maoz**), emit electromagnetic waves in the visible spectrum and can therefore be detected by an optical telescope, the majority of pulsar emission is invisible to the human eye and requires a radio telescope to be detected.

1.3.1 What is a Pulsar Profile?

1.3.2 What is multi-path scattering?

1.3.3 What is a Pulsar Dispersion Measure?

1.3.4 What is a Radio Telescope and how do they work?

1.3.5 What is the Murchison Widefield Array?

1.3.6 Why are we conducting sky surveys?

1.3.7 How is a sky survey conducted with the Murchison Widefield Array?

1.3.8 What is tied-array beamforming?

1.3.9 How many Pulsar candidates are found in a Murchison Widefield Array sky survey?

1.3.10 How long do sky surveys take?

1.3.11 How much data is created from a sky survey?

1.3.12 What is Radio Frequency Interference?

1.3.13 What is a Signal-to-Noise Ratio?

1.3.14 What is PRESTO and a .PFD file?

1.3.15 What is Machine Learning?

1.3.16 How does Machine Learning work?

1.3.17 Why do we need to use Machine Learning in finding new Pulsars?

1.3.18 Why is this particular Machine Learning Classifier used?

2 Methods

2.1 Developing the Machine Learning Training Dataset

Before a machine learning algorithm can make predictions and classify candidates as a pulsar or a non-pulsar, it must first build a classification model from a training dataset which contains similar data with known positive and negative classifications (**tan**; **lyon**). For the use case of pulsar classification, the training dataset must contain examples of data from both pulsars and from non-pulsars so that the algorithm can learn how to distinguish between the two classes.

To maximise the accuracy of the machine learning algorithm, the input data (including the training dataset) must be composed of a common group of features that can be determined for each candidate that maximises the differences between a pulsar and a non-pulsar. The candidate features used by **tan** to maximise the differences between pulsar and non-pulsar candidates are:

$$Prof_{\mu}, Prof_{\sigma}, Prof_{S}, Prof_{k} \tag{1}$$

$$DM_{\mu}, DM_{\sigma}, DM_{S}, DM_{k}, DM_{\mu'}, DM_{\sigma'}, DM_{|S'|}, DM_{k'} \tag{2}$$

$$Subband_{\mu}, Subband_{\sigma}, Subband_{S}, Subband_{k} \tag{3}$$

$$Subint_{\mu}, Subint_{\sigma}, Subint_{S}, Subint_{k} \tag{4}$$

Where candidate features are calculated from the: (1) Integrated Pulsar Profile, (2) the Dispersion Measure – Signal-to-Noise Ratio Curve (DM-S/N), (3) the correlation coefficients between each sub-band and the integrated pulsar profile, and from the (4) correlation coefficients between each sub-integration and the integrated pulsar profile. See *Appendix 2A* for formulae to calculate each feature.

To extract the 20 above features from each classification candidate, we can use the software `PulsarFeatureLab` (**lyon**).

2.1.1 Pulsar Candidate Feature Extraction

The Python software tool `PulsarFeatureLab` can be used to consume pulsar candidate files of the PRESTO Prepfold `PFD` filetype and output the 20 above features for each candidate into a single file of WEKA Data Mining `ARFF` filetype (**lyon**).

To create a closed software environment in which the dependencies of the `PulsarFeatureLab` software are unaffected by the host operating system, a containerised virtual operating system can be created using the free software Docker (`https://docker.com`).

First, a directory to store the Dockerfile and pulsar candidate data is created by completing the following commands in a UNIX terminal:

```
$ mkdir ~/pulsars
$ cd ~/pulsars
$ touch Dockerfile
```

To create the Docker image, the contents of the `Dockerfile` can be edited to contain:

Dockerfile

```
1  FROM alpine/git:latest as builder
2  WORKDIR /root/
3  RUN cd /root/ && git clone --single-branch --branch V1.3.2
       https://github.com/scienceguyrob/PulsarFeatureLab.git &&
       mkdir PulsarFeatureLab/PulsarFeatureLab/Data/IO
4
5  FROM python:2.7
6  WORKDIR /usr/src/app
7  COPY --from=builder /root/PulsarFeatureLab .
8  RUN pip install numpy scipy matplotlib astropy
9  ENTRYPOINT ["python", "./PulsarFeatureLab/Src/
       PulsarFeatureLab.py"]
```

The above `Dockerfile` instructs Docker to:

i. use an image of Alpine Linux with `git` preinstalled to download the `PulsarFeatureLab` software from GitHub (`https://github.com/scienceguyrob/PulsarFeatureLab`);

ii. create a directory inside the downloaded software to store the input and output data;

iii. create a Docker image based on Python 2.7;

iv. transfer the `PulsarFeatureLab` software into the Python 2.7 image; and

v. install `PulsarFeatureLab`'s library dependencies (`NumPy, SciPy, matplotlib` and `astropy`).

The above Docker image can now be built into a container (a virtual operating system) and a directory to hold the input data can be created by running the following commands on a UNIX terminal:

```
$ docker build -t jacobianm/pulsarfeaturelab:1.3.2 .
$ mkdir ~/pulsars/data/pfd
```

Candidate `PFD` files of known pulsars and non-pulsars detected by the Murchison Widefield Array (MWA) provided by N. Swainston can now populate the above created directory, and the following command can be ran to extract the features from the candidates:

```
$ docker run --rm -v ~/pulsars/data/pfd:/usr/src/app/
    PulsarFeatureLab/Data/IO jacobianm/pulsarfeaturelab:1.3.2
     -d "/usr/src/app/PulsarFeatureLab/Data/IO" -c 3 -t 6 -f
    "/usr/src/app/PulsarFeatureLab/Data/IO/output.arff" --
    arff --meta
```

This function instructs Docker to connect the directory containing the `PFD` files to the `PulsarFeatureLab` container's input/output directory and then run the `PulsarFeatureLab` software with arguments stating where the input files are, what filetype they are (`PFD`), which set of features to extract, and where to place the output file.

2.1.2 Creating the Training Dataset

The `output.arff` file created by `PulsarFeatureLab`, contains a set of comma-separated features for each candidate, with an appended '?' character, per file line. Since the class of each candidate is already known, the '?' character on each line can be replaced by a '1' if the candidate is a pulsar, or a '0' if the candidate is a non-pulsar. This signals to the Machine Learning algorithm what a pulsar and a non-pulsar candidate's feature set may appear like.

The edited file can now be renamed and moved with the following command:

```
$ mv ~/pulsars/data/pfd/output.arff ~/pulsars/data/
    trainingSet.arff
```

The Machine Learning Training Dataset has now been created using Murchison Widefield Array data.

2.2 Evaluating the LOTAASClassifier Machine Learning Classification Tool for use with the Murchison Widefield Array

To address the Candidate Selection Problem discussed in *Section 1.3* of this report, a Java Machine Learning pulsar classification tool named `LOTAASClassifier` was created to classify the pulsar candidates produced by the LOFAR Tied-Array All-sky Survey (LOTAAS) (**lyon**). Due to the similarities between the LOFAR radio telescope and the Murchison Widefield Array (MWA) radio telescope, it is logical to attempt to apply the `LOTAASClassifier` tool to candidates produced by the MWA.

The `LOTAASClassifier` tool contains four machine learning algorithms with which the user can choose to make pulsar classifications. They are: the J48 (C4.5 Decision Tree (**quinlan**)), the Multi-Layer Perceptron, the Naive Bayes, and the Support Vector Machines (SVM) algorithms (**lyon**).

To begin evaluating the software, `LOTAASClassifier v1.0` can be downloaded from its GitHub repository

(`https://github.com/scienceguyrob/LOTAASClassifier`) with the following UNIX terminal commands:

```
$ cd ~/pulsars
$ git clone https://github.com/scienceguyrob/LOTAASClassifier
    .git
```

The tool's executable program can then be found by navigating to the following directory in the UNIX terminal:

```
$ cd ~/pulsars/LOTAASClassifier/dist
```

2.2.1 Creating a Classification Model

In order to use `LOTAASClassifier` to classify Murchison Widefield Array candidates, we must use the Machine Learning Training Dataset created in *Section 2.1* to create a classification model. This can be completed by running the following commands:

```
$ java -jar LOTAASClassifier.jar -t ~/pulsars/data/
    trainingSet.arff -m ~/pulsars/data/model.m -a 1
```

This instructs `LOTAASClassifier` to create a new classification model for the J48 machine learning algorithm with the previously constructed training dataset.

To ensure that the classification model was created successfully, we can test the model against the same dataset provided by N. Swainston to create the training dataset. Using `PulsarFeatureLab` we can produce a new `output.arff` file with the candidates' class unedited.

```
$ docker run --rm -v ~/pulsars/data/pfd:/usr/src/app/
    PulsarFeatureLab/Data/IO jacobianm/pulsarfeaturelab:1.3.2
     -d "/usr/src/app/PulsarFeatureLab/Data/IO" -c 3 -t 6 -f
    "/usr/src/app/PulsarFeatureLab/Data/IO/output.arff" --
    arff --meta
```

It is now possible to test the classification model with the candidate feature sets with the following commands:

```
$ cd ~/pulsars/LOTAASClassifier/dist
$ java -jar LOTAASClassifier.jar -p ~/pulsars/data/pfd/output
    .arff -m ~/pulsars/data/model.m -a 1
```

The `LOTAASClassifier` tool will output two files: '`output.positive`' and '`output.negative`', in the same location as the input dataset. The J48 classification model was constructed successfully if the candidates inside the '`positive`' file are the known pulsars.

2.2.2 Evaluating LOTAASClassifier

With a successfully created machine learning classification model, it is now possible to test `LOTAASClassifier` against a previously unseen (by the classifier) dataset of Murchison Widefield Array candidates whose class is also known. We can first remove the existing candidates from inside the `PFD` files directory and delete the existing `output` files.

The candidates directory can then be populated by new `PFD` files and we can use the '`docker run`' command from *Section 2.2.1* to extract the features from the new candidates.

The `LOTAASClassifier` tool can now make classification predictions on the new candidates by running the UNIX terminal command:

```
$ cd ~/pulsars/LOTAASClassifier/dist
$ java -jar LOTAASClassifier.jar -p ~/pulsars/data/pfd/output
    .arff -m ~/pulsars/data/model.m -a 1
```

By inspecting the created '`output.positive`' and '`output.negative`' files and comparing the results to the known pulsars, we can evaluate the performance of the `LOTAASClassifier` tool with Murchison Widefield Array data and make a judgement about further use of the classifier.

2.2.3 Creating the PulsarClassifier Ensemble Classification Tool

According to **tan**, using Machine Learning algorithms in ensemble to make pulsar classifications increases the accuracy of classifications, classifying pulsars that were often misclassified such as wide-pulse pulsars.

To build the ensemble classification feature into the existing `LOTAASClassifier` tool, we can first begin by creating a new Java project named `PulsarClassifier` using the free software, Maven (`https://maven.apache.org`).

```
$ cd ~/pulsars
$ mkdir PulsarClassifier && cd PulsarClassifier
$ mvn archetype:generate -DarchetypeGroupId=org.apache.maven.
    archetypes -DarchetypeArtifactId=maven-archetype-
    quickstart -DarchetypeVersion=1.4
```

We can now copy the source code from `LOTAASClassifier` to be included in the `PulsarClassifier` software.

```
$ cp -R ~/pulsars/LOTAASClassifier/src/ ~/pulsars/
    PulsarClassifier/src/main/java
```

To use the WEKA suite of Machine Learning tools, we must then edit the `pom.xml` file inside `PulsarClassifier` to include it as a dependency, and resynchronise the project:

pom.xml

```
...
<dependencies>
...
    <dependency>
        <groupId>nz.ac.waikato.cms.weka</groupId>
        <artifactId>weka-stable</artifactId>
        <version>3.8.0</version>
    </dependency>
...
</dependencies>
...
```

We now have the following basic project directory structure:

```
PulsarClassifier/
    src/
        main/
            java/
                com/jacobianmatthews/pulsarclassifier/
                com/scienceguyrob/lotaasclassifier/
        test/
            java/
                com/jacobianmatthews/pulsarclassifier
    target/
            ...
    pom.xml
```

Where the new source code will be located under `/src/main/java/com/jacobianmatthews/pulsarclassifier`. To introduce the ensemble classification feature, we must write four main Java classes: `PulsarClassifier.java`, `ClassifierBuilder.java`, `ClassifierValidator.java`, and `ClassPredictor.java`.

The `LOTAASClassifier` tool accepts a command-line argument `-a` which accepts an integer that denotes the machine learning algorithm to use in building a classification model and making predictions (**lyon**). Therefore, we will add an algorithm into the above listed Java classes that will accept an integer value of `-1` that will activate the ensemble classifier.

To add ensemble classification to the class `ClassifierBuilder.java`, we will use the following algorithm:

2.2.4 Evaluating PulsarClassifier

3 Results and Outputs

3.1 Machine Learning Training Dataset

The machine learning training dataset created with candidates detected by the Murchison Widefield Array can be found under *Appendix 8.3*.

3.2 Output from LOTAASClassifier with Murchison Widefield Array Data

3.3 Output from PulsarClassifier Machine Learning Classification Tool

4 Discussion

4.1 How effective are the training datasets?

4.2 Evaluating Curtin Institute of Radio Astronomy's Pulsar Classification Pipeline

4.3 Why did/didn't the LOFAR Machine Learning Algorithm work with the MWA?

4.4 What changes to the algorithm were necessary for it to successfully classify Pulsars from the MWA?

5 Conclusions

6 Recommendations

7 References

8 Appendices

8.1 Pulsar Feature Lab

8.2 Pulsar Classifier

8.3 Machine Learning Training Dataset

trainingData.arff

```
@relation Pulsar_Feature_Data_Type_6
@attribute Feature_1 numeric
@attribute Feature_2 numeric
@attribute Feature_3 numeric
@attribute Feature_4 numeric
@attribute Feature_5 numeric
@attribute Feature_6 numeric
@attribute Feature_7 numeric
@attribute Feature_8 numeric
@attribute Feature_9 numeric
@attribute Feature_10 numeric
@attribute Feature_11 numeric
@attribute Feature_12 numeric
@attribute Feature_13 numeric
@attribute Feature_14 numeric
@attribute Feature_15 numeric
@attribute Feature_16 numeric
@attribute Feature_17 numeric
@attribute Feature_18 numeric
@attribute Feature_19 numeric
@attribute Feature_20 numeric
@attribute class {0,1,2}
@data
105.43918879794592,34.81562200057764,0.64091523062507,2.999965982775474,4.9731636,0.55617476,0.73770590325867
    usr/src/app/PulsarFeatureLab/Data/IO/1222697776_DM106.85
    _ACCEL_0_2_23_29_01.45_-10_35_14.03_8577.99ms_Cand_rfi.pfd
19.439267094649395,37.83110112790925,5.311899765657869,27.72625393618159,49.398266,17.023054,0.75823017779194
    usr/src/app/PulsarFeatureLab/Data/IO/1252780888_100_bins_PSR_0152-1637.
    pfd
18.18808199226157,29.34228539923202,5.972780846625737,41.45065223740317,31.696558,10.691088,0.632813129945887
    usr/src/app/PulsarFeatureLab/Data/IO/1255197408_100_bins_PSR_0459-0210.
    pfd
21.22497781504873,33.50944234920476,4.928435748454679,26.392961286010472,19.536982,9.97172,1.0080046130377358
    usr/src/app/PulsarFeatureLab/Data/IO/1253471952_100_bins_PSR_0255-5304.
    pfd
125.23524447122634,56.134362050023924,0.2229079772063587,-0.6216367448361071,11.638162,0.4708962,-0.932350032
    usr/src/app/PulsarFeatureLab/Data/IO/1222697776_DM13.36
    _ACCEL_0_5_23_40_05.66_-13_34_55.98_591.37ms_Cand_noise.pfd
89.58174363355785,48.4446505060987,1.1788897000809406,2.1990474951218184,6.3838625,1.0415499,0.57757212475454
    usr/src/app/PulsarFeatureLab/Data/IO/1252177744_100_bins_PSR_0151-0635.
    pfd
24.10863128259143,28.805247892957368,5.8820714970090355,41.4163908926057,26.94988,5.6278524,0.587453916304551
    usr/src/app/PulsarFeatureLab/Data/IO/1253991112_100_bins_PSR_0206-4028.
    pfd
19.659448914180896,33.71258916374586,6.212422938671684,38.92028743782727,33.78987,11.8825655,0.61393884706171
    usr/src/app/PulsarFeatureLab/Data/IO/1255803168_100_bins_PSR_0401-7608.
    pfd
130.37194427538753,48.5719656104399,0.03110721286401433,-0.28624231198914396,3.4748905,0.32001108,0.544493918
    usr/src/app/PulsarFeatureLab/Data/IO/1222697776_DM116.10
    _ACCEL_0_6_23_42_17.97_-12_35_31.39_3956.39ms_Cand_rfi.pfd
14.60760000762955,33.37521977816117,5.2251889142177985,29.534224184342584,80.282196,26.19628,0.51782067247501
    usr/src/app/PulsarFeatureLab/Data/IO/1255197408_100_bins_PSR_J0450-1248.
    pfd
```

```
19.160516821154644,52.612510321908914,3.498625248210871,10.92536155890088,221.91383,63.068035,0.4634763573806
    usr/src/app/PulsarFeatureLab/Data/IO/1255197408_100_bins_PSR_0452-1759.
    pfd
105.40739512137863,42.046982247914656,1.105401957592074,2.71606611188946,9.959542,1.508026,-0.107286435139737
    usr/src/app/PulsarFeatureLab/Data/IO/1222697776_DM124.60
    _ACCEL_0_5_23_30_05.03_-15_32_28.86_7306.99ms_Cand_rfi.pfd
108.62704623126228,48.50808371245384,0.3888816698242244,0.10828707118695347,3.2382581,0.57282144,0.5061148662
    usr/src/app/PulsarFeatureLab/Data/IO/1222697776_DM144.10
    _ACCEL_0_4_23_41_42.76_-10_20_02.01_8566.62ms_Cand_rfi.pfd
30.65638094885709,36.39390906613557,4.334831666786916,20.661559138936177,22.137678,7.129584,0.583874112231887
    usr/src/app/PulsarFeatureLab/Data/IO/1254594264_100_bins_PSR_0304+1932.
    pfd
110.46206868842188,46.251025957184034,0.37602640993908826,0.6004748040886194,3.5859008,0.48099223,0.954763129
    usr/src/app/PulsarFeatureLab/Data/IO/1222697776_DM133.85
    _ACCEL_0_4_23_34_32.65_-11_35_39.59_6359.79ms_Cand_rfi.pfd
67.96984095457957,30.670603367741087,2.9112873304761235,15.54008032933561,10.704723,3.0320911,-0.031139309413
    usr/src/app/PulsarFeatureLab/Data/IO/1222697776_DM119.85
    _ACCEL_0_2_23_29_31.44_-15_47_03.39_13154.44ms_Cand_rfi.pfd
118.82100466581987,47.54490692614183,0.07781892865435884,-0.01638792599233385,3.4399548,0.8459567,0.703248063
    usr/src/app/PulsarFeatureLab/Data/IO/1222697776_DM137.35
    _ACCEL_0_6_23_43_56.97_-11_20_36.48_5794.50ms_Cand_rfi.pfd
106.55367062734199,38.31922288316345,1.1201921873325247,3.524621849432644,4.6642995,1.4868807,1.2001879564459
    usr/src/app/PulsarFeatureLab/Data/IO/1222697776_DM128.85
    _ACCEL_0_2_23_35_06.34_-10_20_02.01_13136.61ms_Cand_rfi.pfd
110.58047755262913,56.835988222404985,0.2597416198749472,-0.6009060312856693,6.2209997,1.3479991,0.0301769380
    usr/src/app/PulsarFeatureLab/Data/IO/1222697776_DM124.10
    _ACCEL_0_12_23_36_13.35_-15_17_52.85_590.99ms_Cand_noise.pfd
22.63765924905709,32.25698418159722,5.017978110756053,28.558061328289384,23.729832,7.0639696,0.65822770848595
    usr/src/app/PulsarFeatureLab/Data/IO/1256407632_100_bins_PSR_J0450-1248.
    pfd
113.87621446459325,43.41151017852675,0.5893561441860531,1.71756136954156,5.65921,1.1683735,1.0193316046807013
    usr/src/app/PulsarFeatureLab/Data/IO/1222697776_DM119.35
    _ACCEL_0_2_23_29_35.58_-09_49_30.86_10373.16ms_Cand_rfi.pfd
116.41310084274251,46.52577061051363,0.1525172365016325,0.14280165135512535,3.0154004,0.65890557,1.4567341719
    usr/src/app/PulsarFeatureLab/Data/IO/1222697776_DM106.85
    _ACCEL_0_2_23_37_20.08_-15_17_52.85_9862.48ms_Cand_rfi.pfd
115.05147692422577,59.943352711870936,0.06373023728528525,-0.5571994750116955,5.8320494,0.6752133,0.100135170
    usr/src/app/PulsarFeatureLab/Data/IO/1222697776_DM114.60
    _ACCEL_0_6_23_42_50.44_-11_20_36.48_3514.62ms_Cand_rfi.pfd
13.688631621147106,29.885241623656917,6.142618064042354,42.30058250077879,42.656235,14.500364,0.6562798650382
    usr/src/app/PulsarFeatureLab/Data/IO/1256407632_100_bins_PSR_0459-0210.
    pfd
```