

IDC I - Final Project Report

Identifying Pulsar

Candidates

(Pulsar classification)

Submitted to -

Dr. P Radha Krishna, Professor, Department of CSE,
NIT, Warangal-506004, Telangana, INDIA

Submitted by -

Adarsh Kumar Dash
17003, 4th year BS-MS Dual Degree,
IISER Berhampur
22nd December, 2020



Abstract

In this project, the machine learning algorithms Logistic Regression, k- nearest neighbours, Decision Tree Classification and Random Forest Classification models have been used to predict if a given stellar object is a Pulsar or not, depending on a number of properties of the same.

The four models used have been compared to see which gives better results and lower errors in the required prediction. The dataset used here is taken from **HTRU2 pulsar dataset**.

The algorithms used here were referred from many different sources available on the internet. The software implementations that were used in this project were taken from Scikit-learn python library.

Introduction

Pulsar classification is a great example of where machine learning can be used beneficially in astrophysics.

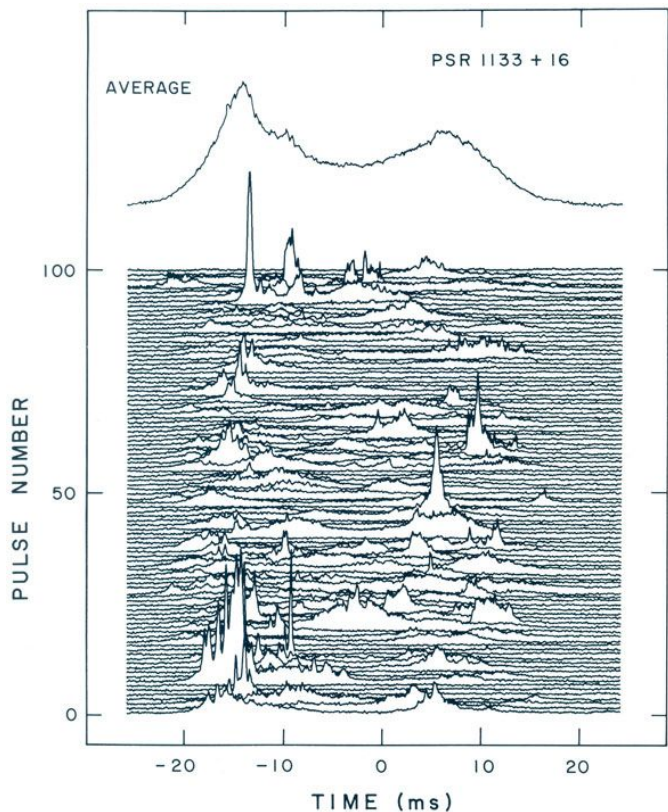
Pulsars are “pulsating radio sources”, now known to be caused by rapidly rotating neutron stars. [Neutron stars](#) are the relics of dead massive stars, they’re small and extremely dense. A characteristic property of pulsars are the periodic bursts of emission produced by their radio emitting jets. As the pulsar rotates, the direction of this emission also rotates and astronomers see a pulse of radio emission each time one of the jets points towards the Earth.


In order to classify a data sample as a pulsar or not a pulsar, we need to be able to extract some information on the data sample that can characterise its class. The individual bursts of emission from a pulsar (i.e. the pulses) do not have a constant shape or amplitude, so individually they’re not very useful for uniquely identifying a pulsar.

Because the individual pulses are all different, astronomers stack them up and create an average integrated pulse profile to characterise a particular pulsar as shown in the figure.

Additionally the pulse will arrive at different times across different radio frequencies. The delay from frequency to frequency is caused by the ionised inter-stellar medium and is known as the dispersion.

Astronomers fit for the shape of the delay in order to compensate for its effect, but there’s always an uncertainty associated with the fit. That is expressed in the DM-SNR (“dispersion-measure-signal-to-noise-ratio”) curve. When these two curves are put together, we see for each pulsar candidate there are eight numerical characteristic features that can be extracted as standard: four from the integrated pulse profile and four from the DM-SNR curve.





They are :

1. Integrated (Intensity) profile mean
2. Integrated (Intensity) profile standard deviation
3. Integrated (Intensity) profile kurtosis
4. Integrated (Intensity) profile skewness
5. DM-SNR curve mean
6. DM-SNR curve standard deviation
7. DM-SNR curve kurtosis
8. DM-SNR curve skewness

Objective

To measure how accurately the machine learning models predict if a given candidate is a pulsar based on the HTRU2 pulsar dataset available in the Kaggle using four machine learning algorithms :

1. Logistic regression
2. KNN classification
3. Decision tree classification
4. Random forest classification

Background

Data science uses scientific methods, processes, algorithms and systems to extract knowledge and insights from many structural and unstructured data. It is also rapidly growing with many advancements in recent years; most notably within Machine Learning.

Machine Learning is the study of algorithms that can learn from the data and can make predictions. These algorithms are mainly categorised into :

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning

Logistic Regression

Logistic Regression is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.). In other words, the logistic regression model predicts $P(Y=1)$ as a function of X .


Logistic Regression Assumptions

1. Binary logistic regression requires the dependent variable to be binary.
2. For a binary regression, the factor level 1 of the dependent variable should represent the desired outcome.
3. Only the meaningful variables should be included.
4. The independent variables should be independent of each other. That is, the model should have little or no multicollinearity.
5. The independent variables are linearly related to the log odds.
6. Logistic regression requires quite large sample sizes.

KNN Classification

KNN (K-Nearest Neighbor) is a simple supervised classification algorithm we can use to assign a class to new data point. It can be used for regression as well, KNN does not make any assumptions on the data distribution, hence it is non-parametric. It keeps all the training data to make future predictions by computing the similarity between an input sample and each training instance.

KNN can be summarized as below:

- 
1. Computes the distance between the new data point with every training example.
 2. For computing the distance measures such as Euclidean distance, Hamming distance or Manhattan distance will be used.
 3. Model picks K entries in the database which are closest to the new data point.
 4. Then it does the majority vote i.e the most common class/label among those K entries will be the class of the new data point.

Decision tree and random forest classification

Decision trees are a type of model used for both classification and regression. Trees answer sequential questions which send us down a certain route of the tree given the answer. The model behaves with “if this than that” conditions ultimately yielding a specific result.

Advantages to using decision trees:

1. Easy to interpret and make for straightforward visualizations.
2. The internal workings are capable of being observed and thus make it possible to reproduce work.
3. Can handle both numerical and categorical data.
4. Perform well on large datasets
5. Are extremely fast

Disadvantages of decision trees:

1. Building decision trees require algorithms capable of determining an optimal choice at each node.
2. Decision trees are prone to overfitting, especially when a tree is particularly deep.

Ideally, we would like to minimize both error due to bias and error due to variance. Enter random forests. Random forests mitigate this problem well. A random forest is simply a collection of decision trees whose results are aggregated into one final result. Their ability to limit overfitting without substantially increasing error due to bias is why they are such powerful models.

One way Random Forests reduce variance is by training on different samples of the data. A second way is by using a random subset of features. Random forests are a strong modeling technique and much more robust than a single decision tree. They aggregate many decision trees to limit overfitting as well as error due to bias and therefore yield useful results.



Dataset Description

The data set has to be taken large enough to get good training and predictions. So it was chosen to be HTRU2 pulsar data set. It has 8 features and 1 class column. The columns are named as 'mean_int_pf', 'std_pf', 'ex_kurt_pf', 'skew_pf', 'mean_dm', 'std_dm', 'kurt_dm', 'skew_dm', 'class'.

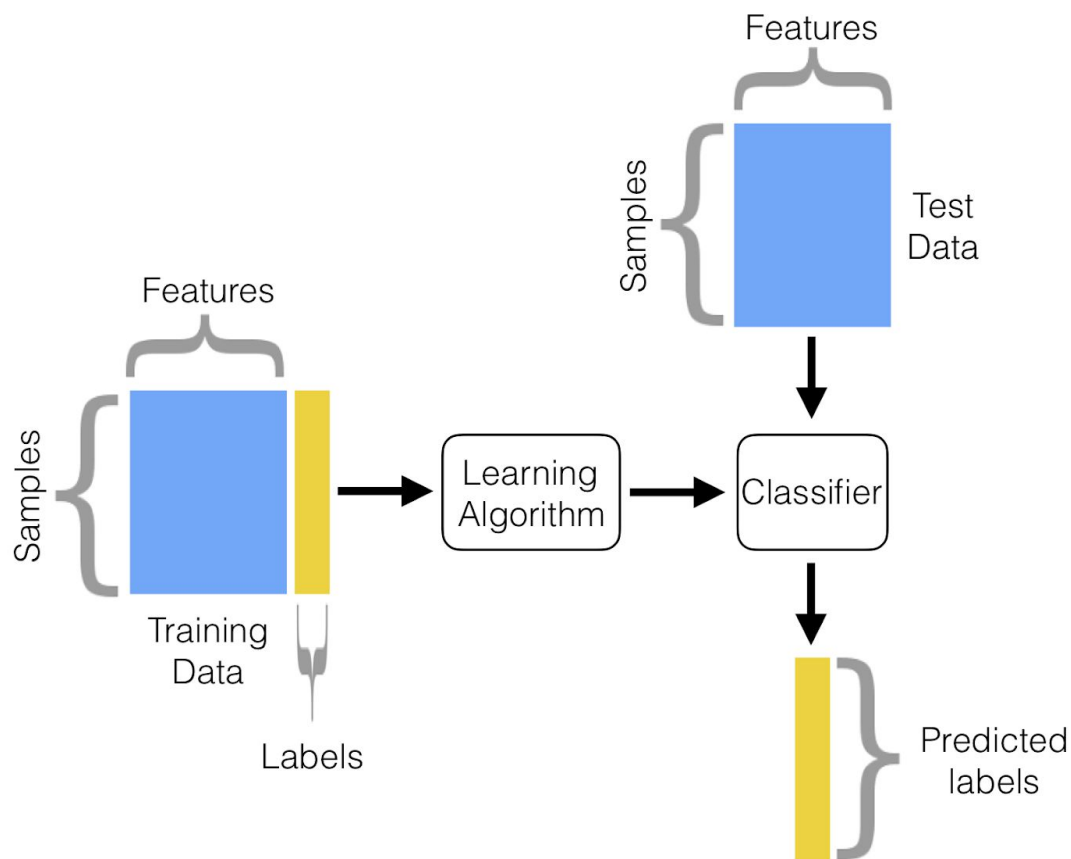
The original dataset after downloading had to be edited a little to add the labels/heads to the columns, as guided by the dataset downloading site.

All of these are float64 variables, except the column 'class' which takes int64 variables. If the value is a 1 - it is a pulsar - if 0 - it is not.

There are a total of 17,897 entries,

Implementation

The basic flow chart of the whole program is shown below :



The training data set is split into two parts such that one part is used for training and the other part (smaller) is used for prediction. Then the predicted values are compared with the original ones.

Depending on that, accuracy is calculated and a confusion matrix is made.

Code snippets

The following libraries were used for the purpose.

```
# Import the NumPy and Pandas packages

import numpy as np
import pandas as pd

#Import matplotlib and seaborn for visualizations

import matplotlib.pyplot as plt
import seaborn as sns
from google.colab import files

uploaded = files.upload()
```

The header list was updated for naming each column as per the guidelines in the source of dataset download. The details of each label is also shown below.

```
header_list = ["mean_int_pf", "std_pf", "ex_kurt_pf", "skew_pf", "mean_dm", "std_dm", "kurt_dm", "skew_dm", "class"]
data = pd.read_csv('HTRU_2.csv', names=header_list)
```

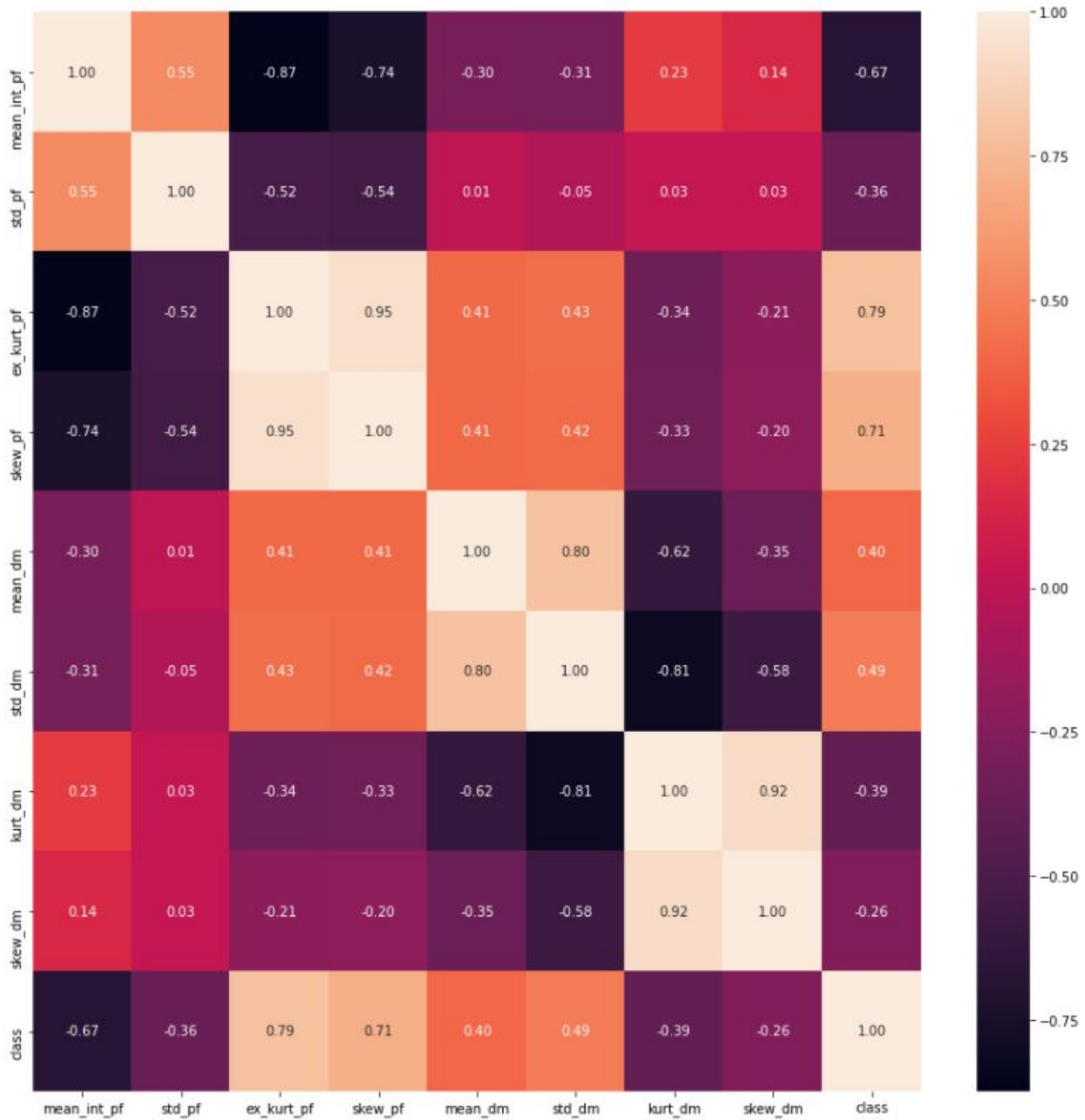
```
data.head()
```

	mean_int_pf	std_pf	ex_kurt_pf	skew_pf	mean_dm	std_dm	kurt_dm	skew_dm	class
0	140.562500	55.683782	-0.234571	-0.699648	3.199833	19.110426	7.975532	74.242225	0
1	102.507812	58.882430	0.465318	-0.515088	1.677258	14.860146	10.576487	127.393580	0
2	103.015625	39.341649	0.323328	1.051164	3.121237	21.744669	7.735822	63.171909	0
3	136.750000	57.178449	-0.068415	-0.636238	3.642977	20.959280	6.896499	53.593661	0
4	88.726562	40.672225	0.600866	1.123492	1.178930	11.468720	14.269573	252.567306	0

```
data.info()
```

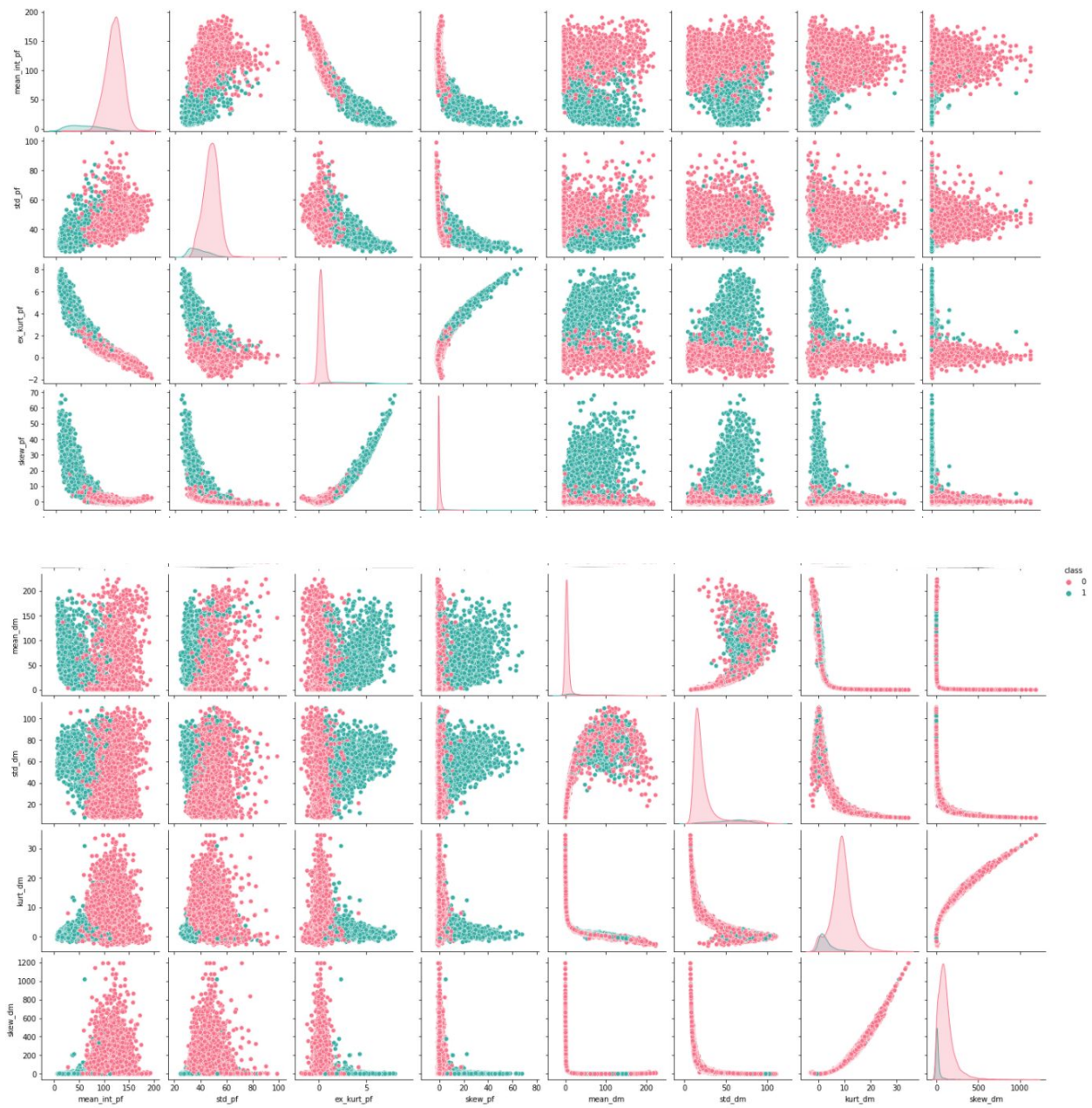
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17898 entries, 0 to 17897
Data columns (total 9 columns):
#   Column      Non-Null Count  Dtype
---  -
0   mean_int_pf  17898 non-null  float64
1   std_pf       17898 non-null  float64
2   ex_kurt_pf   17898 non-null  float64
3   skew_pf      17898 non-null  float64
4   mean_dm      17898 non-null  float64
5   std_dm       17898 non-null  float64
6   kurt_dm      17898 non-null  float64
7   skew_dm      17898 non-null  float64
8   class        17898 non-null  int64
dtypes: float64(8), int64(1)
memory usage: 1.2 MB
```

For further analysis of the given dataset, a heat-map was made to check what is the **correlation** between each feature as shown below :



It can be observed that the most correlated features are mean_dm - std_dm and kurt_dm - skew_dm. Furthermore, we may infer that ex_kurt_pf has the highest effect on the value of the class (either 1 or 0).

Similar dependency can be seen graphically too.



After all this analysis, the x_data was scaled and y data was taken as the class. Then scikit_learn library was used further. The dataset was split for training and testing.

```
y = data["class"].values
x_data = data.drop(["class"],axis=1)
x = (x_data - np.min(x_data))/(np.max(x_data)-np.min(x_data))

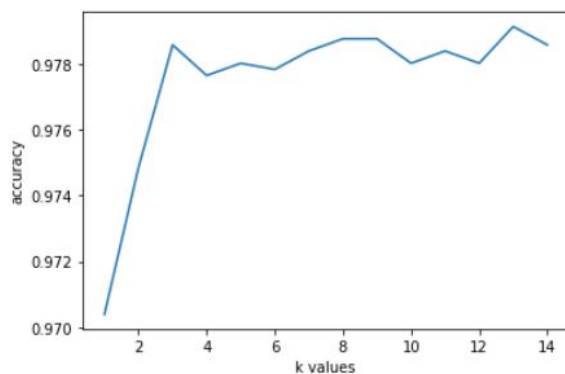
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size = 0.3,random_state=1)
```

Now, the 4 different methods were applied on the modified dataset. The 4 functions typed, returned the values of precision, recall, f1-score and accuracy for each method as follows :

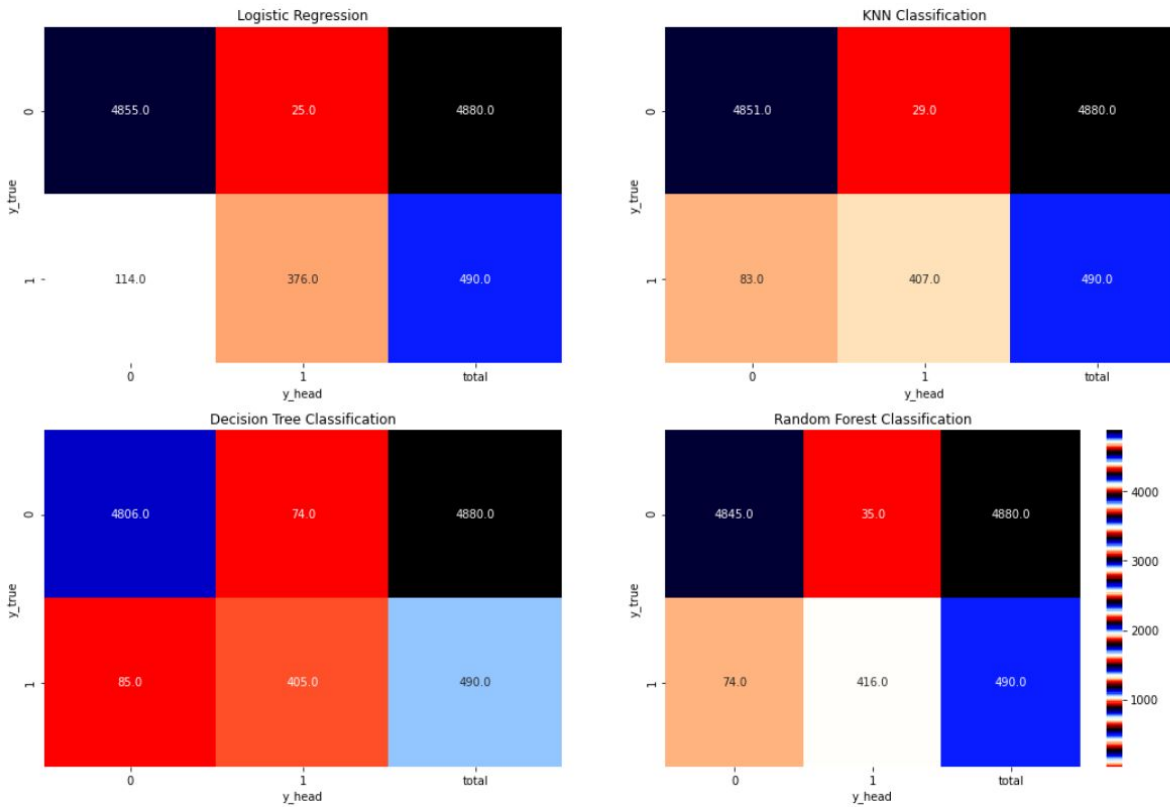
Classification report for Random Forest Classification:				Classification report for Logistic Regression:			
	precision	recall	f1-score		precision	recall	f1-score
0	0.98	0.99	0.99	0	0.98	0.99	0.99
1	0.92	0.85	0.88	1	0.94	0.77	0.84
accuracy	0.98			accuracy	0.97		

Classification report for Decision Tree Classification:				Classification report for KNN Classification:			
	precision	recall	f1-score		precision	recall	f1-score
0	0.98	0.98	0.98	0	0.98	0.99	0.99
1	0.85	0.83	0.84	1	0.93	0.83	0.88
accuracy	0.97			accuracy	0.98		

Along with it, for kNN classification, the dependence of accuracy of prediction on the k-values was also plotted as follows:



To get a clear picture of the prediction by each method, a confusion matrix was plotted for each method.

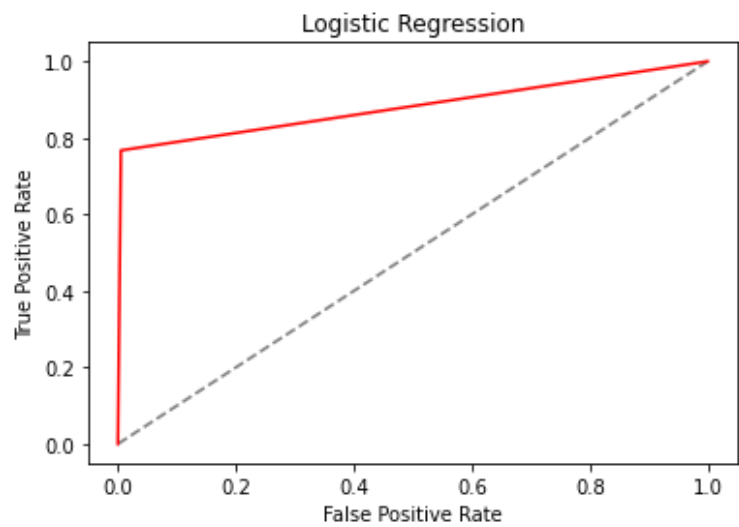
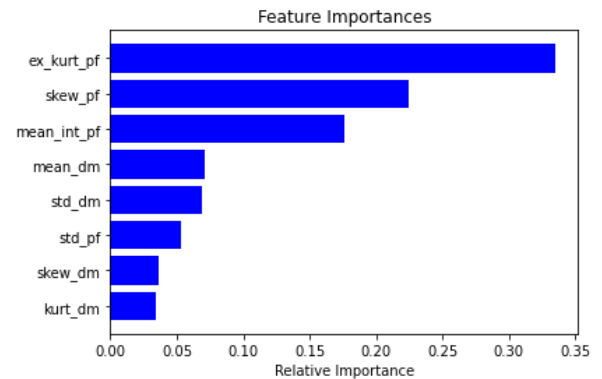


Results and discussion

We can observe that KNN and Random forest classification techniques gave the best accuracy. Still, it can be said that because of the large data set, there is not much difference between the accuracy of each method that was used. Further analysing the results, as guessed using the correlation, we can confirm the importance of each feature in deciding the outcome.

Drawing a ROC curve (for logistic regression), we test that our classifier can be taken as good classifier to the purpose.

We must note that being in nature, the pulsars can have a variety of features that are not predictable directly. Nevertheless, as still they abide by the laws of physics, a classifier with 97% or 98% accuracy can fairly serve the purpose.



References

1. <https://as595.github.io/classification/#pulsars>
2. <https://github.com/topics/htru2>
3. <https://towardsdatascience.com/decision-trees-and-random-forests-df0c3123f991>
4. <https://towardsdatascience.com/scikit-learn-decision-trees-explained-803f3812290d>
5. <https://towardsdatascience.com/knn-using-scikit-learn-c6bed765be75>
6. <https://towardsdatascience.com/building-a-logistic-regression-in-python-step-by-step-becd4d56c9c8>