Individual Project – Supervised Learning

By Tanvi Kini

# Predicting if a star is a Pulsar using Logistic Regression

## Dataset

The dataset used here is called ‘Predicting a Pulsar Star’ and can be found at the following address:

<https://www.kaggle.com/pavanraj159/predicting-a-pulsar-star>

This dataset consists of different attributes of discovered stars.

Pulsars are a rare type of Neutron star that produce radio emission detectable here on Earth. They are of considerable scientific interest as probes of space-time, the inter-stellar medium, and states of matter

Pulsars emit beams as they emit different beams that produce detectable radio emissions. This dataset consists of recordings of different characteristics of stars observed along with some spurious data which will help us identify what characteristics make a pulsar. This dataset has 9 columns and ~18000 rows.

The different column names are as follows:

1. Mean.of.the.integrated.profile - Mean of the integrated profile.

2. Standard.deviation.of.the.integrated.profile - Standard deviation of the integrated profile.

3. Excess.kurtosis.of.the.integrated.profile - Excess kurtosis of the integrated profile.

4. Skewness.of.the.integrated.profile - Skewness of the integrated profile.

5. Mean.of.the.DM.SNR.curve - Mean of the DM-SNR curve.

6. Standard.deviation.of.the.DM.SNR.curve - Standard deviation of the DM-SNR curve.

7. Excess.kurtosis.of.the.DM.SNR.curve - Excess kurtosis of the DM-SNR curve.

8. Skewness.of.the.DM.SNR.curve - Skewness of the DM-SNR curve.

9. Target\_class – Class of the star

The Class or target\_class variable is our response variable. This is binary variable which takes up the value ‘0’ when the star being classified is identified to not be a pulsar and ‘1’ when it is.

I will be using a Logistic Regression Model to predict whether a given star is pulsar based on the data available

# Creating the Logistic Regression Model

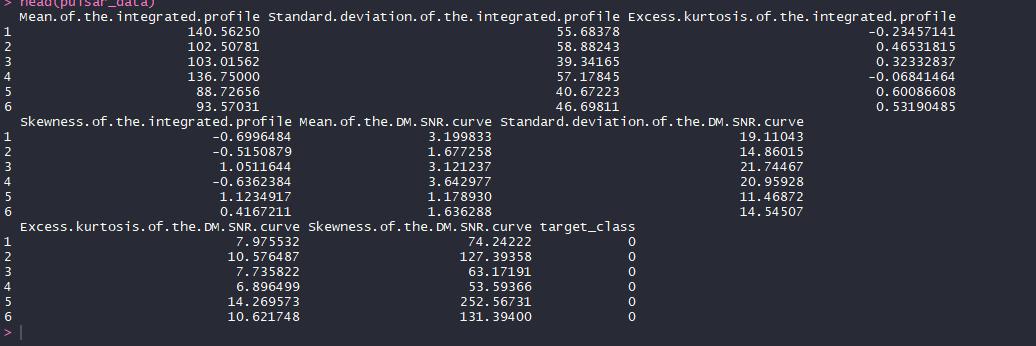
**Reading the data into R**.

Code:

## pulsar\_data <- read.csv("pulsar\_stars.csv")

## head(pulsar\_data)

Output:

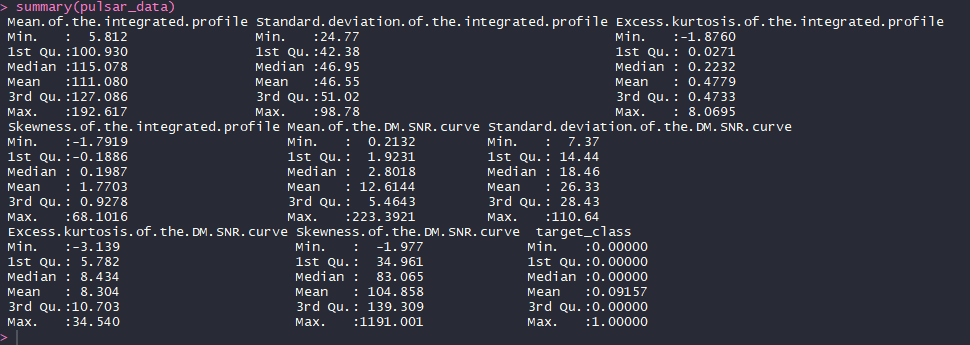


**Checking the summary of our data frame**

Code:

## summary(pulsar\_data)

Output:



**Splitting the data into train and test data**

The data was split into an 80:20 ratio into two separate datasets. Where trainData consists of 80% of the data and testData consists of 20% of the data. I will be using trainData to create the model and testData to test the validity of the model.

Code:

## index <- sample(nrow(pulsar\_data), nrow(pulsar\_data)\*0.80)

## trainData = pulsar\_data[index,]

## testData = pulsar\_data[-index,]

**Creating a logistic regression model**

We can create a logistic regression model using the glm() function as shown below

Code:

## pulsar\_model1 <- glm(trainData$target\_class ~ trainData$Mean.of.the.integrated.profile+

## trainData$Standard.deviation.of.the.integrated.profile+

## trainData$Excess.kurtosis.of.the.integrated.profile+

## trainData$Skewness.of.the.integrated.profile+

## trainData$Mean.of.the.DM.SNR.curve+

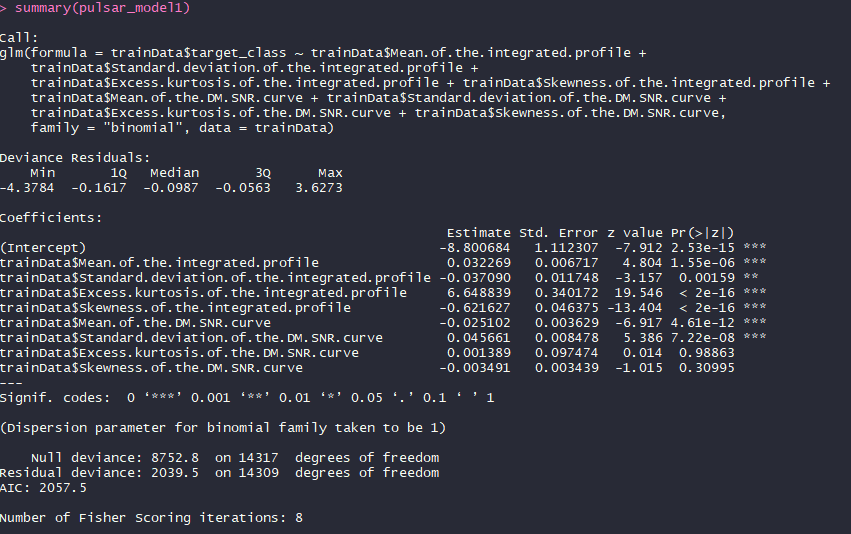
## trainData$Standard.deviation.of.the.DM.SNR.curve+

## trainData$Excess.kurtosis.of.the.DM.SNR.curve+

## trainData$Skewness.of.the.DM.SNR.curve, data = trainData, family = "binomial")

## summary(pulsar\_model1)

Output:



In the summary above, we can see that 2 of the covariates – Excess.kurtosis.of.the.DM.SNR.curve and Skewness.of.the.DM.SNR.Curve have p-values greater than 0.05. This means that these covariates are insignificant and can be eliminated.

**Eliminating the insignificant covariates**

Using the process of backward elimination, I will be eliminating the insignificant covariates and creating a second model

Code:

## pulsar\_model2 <- glm(trainData$target\_class ~ trainData$Mean.of.the.integrated.profile+

## trainData$Standard.deviation.of.the.integrated.profile+

## trainData$Excess.kurtosis.of.the.integrated.profile+

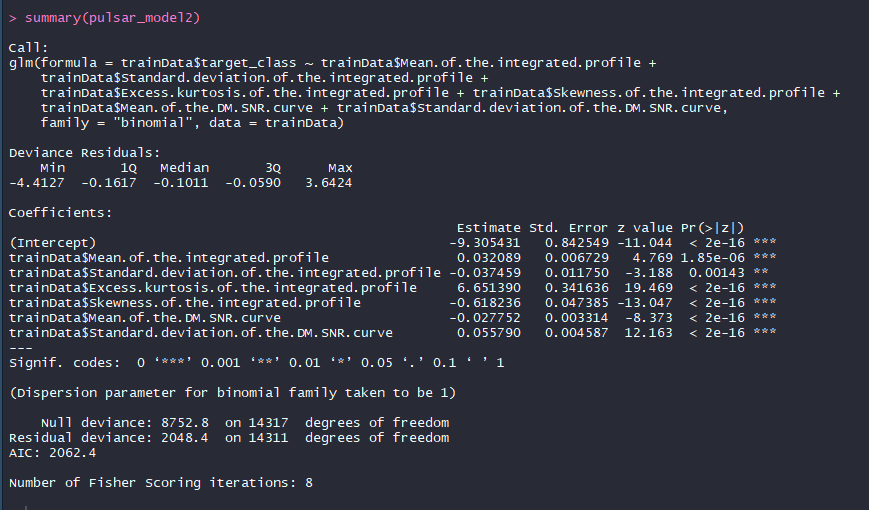
## trainData$Skewness.of.the.integrated.profile+

## trainData$Mean.of.the.DM.SNR.curve+

## trainData$Standard.deviation.of.the.DM.SNR.curve, data = trainData, family = "binomial")

## summary(pulsar\_model2)

Output:



Here we see that pulsar\_model2 has all significant independent variables as all of them have p-values greater than 0.05. We can now proceed to validate the model.

## Model Adequacy Checking

We can use the predict() function to find values for all the data points within the range that our dataset deals with for which we do not have actual observations using the model we created. We can convert a percentage probability into a binary outcome by assuming that any probability greater than 50% i.e greater than 0.5 indicates a value ‘1’ in this case and any probability less than 0.5 indicated a ‘0’.

Code:

## pred\_prob = predict(pulsar\_model2, trainData, type = "response")

## pred\_value = 1\*(pred\_prob>0.5)

## cbind(trainData, pred\_prob, pred\_value)

**Creating a confusion matrix**

A confusion matrix is a table with four combination of predicted and actual values. We can determine which category our predictions fall into.

Following are the possible categories:

1. True-positives: Values that were correctly predicted to be true.

2. False-positives: Values that were false but predicted to be true.

3. True-negatives: Values that were correctly predicted to be negative.

4. False-negatives: Values were true but predicted to be negative.

A confusion matrix was created to check how adequate our logistic regression model was within the trainData.

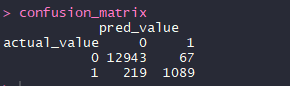
Code:

## actual\_value = trainData$target\_class

## confusion\_matrix = table(actual\_value, pred\_value)

## confusion\_matrix

Output:



## **Calculating misclassification Error rate**

Code:

## misclassification\_error\_rate=1-sum(diag(confusion\_matrix))/sum(confusion\_matrix)

## misclassification\_error\_rate

Output:



We see above that the misclassification error rate is only ~2% which is a good error rate. This shows that our model is actually adequate.

**Plotting a precision recall curve**

When we have a dataset with a binary outcome but a huge disparity in between one kind of response to another we use a precision curve. In our dataset, number of rows with target\_class as 0 is much higher than the number of rows with targer\_class having the value 1. I will therefore plot a precision curve.

Code:

## is\_pulsar = pred\_prob[trainData$target\_class == '1']

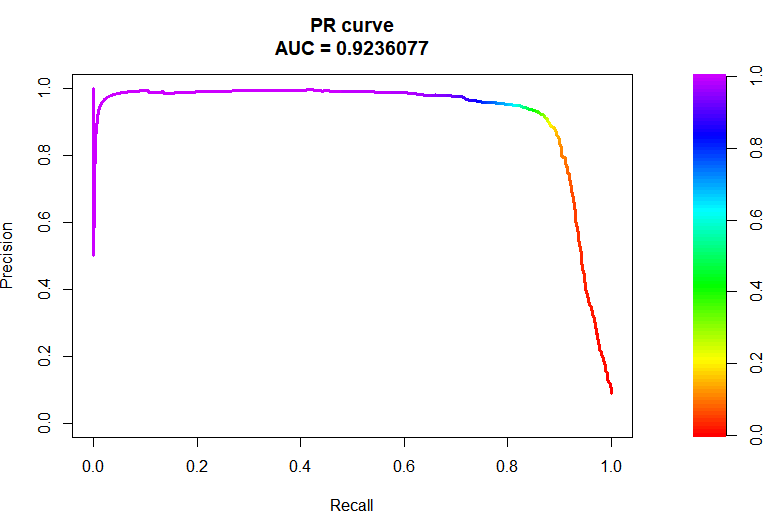
## is\_not\_pulsar = pred\_prob[trainData$target\_class == '0']

## library(PRROC)

## pr = pr.curve(is\_pulsar,is\_not\_pulsar, curve = T)

## plot(pr)

Output:



The curve pictured above shows us that the AUC (Area under the curve) is 0.92. This is a good value and further goes to prove that the model is adequate.

# Working with Test data

Code:

ppt = predict(pulsar\_model2, testData, type = "response")

pvt = 1\*(ppt>0.5)

cbind(testData, ppt, pvt)

**Creating a confusion matrix**

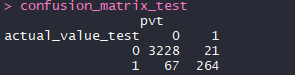
Code:

actual\_value\_test = testData$target\_class

confusion\_matrix\_test = table(actual\_value\_test, pvt)

confusion\_matrix\_test

Output:



**Calculating misclassification Error rate**

Code**:**

misclassification\_error\_rate\_test=1-sum(diag(confusion\_matrix\_test))/sum(confusion\_matrix\_test)

misclassification\_error\_rate\_test

Output:



We can see above that the misclassification error rate for the test data is ~2.4 which is slightly more than that obtained from the trainData. This is still a really good error rate and proves that our model is adequate.

**Plotting the precision recall curve**

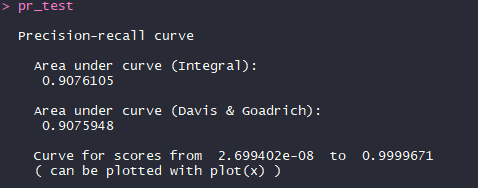
Code:

pr\_test = pr.curve(is\_pulsar\_test,is\_not\_pulsar\_test, curve = T)

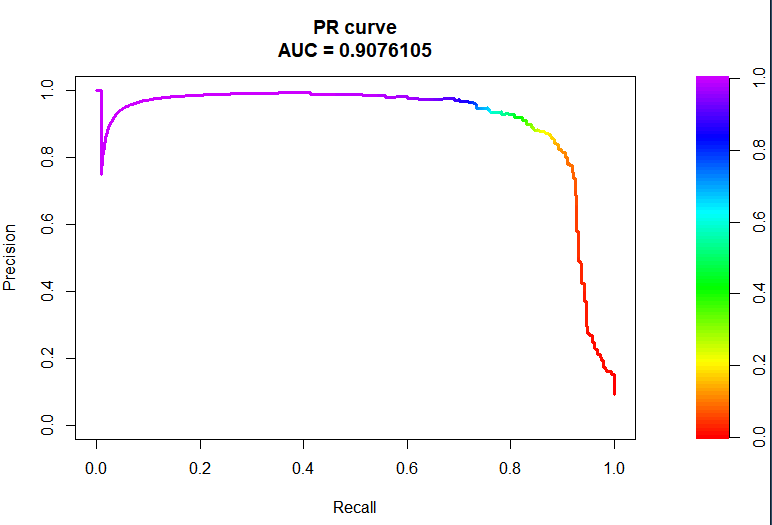
pr\_test

plot(pr\_test)

Output:



PR Curve:



In the PR curve pictured above, we can see that the AUC is 0.90. This is slightly less than that of the trainData and this is expected.

# Conclusion:

We can therefore conclude that the following covariates significantly affect our outcome variable – target\_class.

Mean of the integrated profile.

Standard deviation of the integrated profile.

Excess kurtosis of the integrated profile.

Skewness of the integrated profile.

Mean of the DM-SNR curve.

Standard deviation of the DM-SNR curve.

Using only this information we can accurately predict with a 97.6% accuracy rate whether a given star is a pulsar.