Introduction

The aim of this report is to carry out various classification methods on a dataset and deliver a conclusion towards which method will be the most effective in predicting future values. The data used in this report is taken from a master set of pulsar candidates, where the sample used included 1000 randomly sampled instances of the master dataset. Within the dataset there are a number of variables which relate to the identification of pulsars.

These are:

- 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-SMR curve
- Standard Deviation of the DM-SMR curve
- Excess kurtosis of the DM-SMR curve

The dataset also contains a class designation for each observation which has been determined by human annotators. A class can either be 1 (pulsar) or 0 (noise). In this dataset, 904 observations have been classed as noise, with 96 classed as pulsars. The objective of this report to use the data to select a clear classification method which can be used to determine if future observations can be classed as pulsars or noise.

Method

Splitting Data

To correctly apply different classification methods the 1000 data values were split into three different subsets. These subsets are training, validation and test. This splitting of the data is done to avoid overfitting, where a certain method will be tailor made for the data used to create the classification rule and therefore not useful for prediction of future values. Each of the three subsets have a different function in the method, they are as follows:

- **Training Data**, used to fit the classification model and create classification rules, this is roughly about 50% of the dataset
- **Validation Data**, used to create measures to determine how well a particular method has performed and to create a measure for comparison to other methods. This is roughly about 25% of the dataset
- **Test Data**, used to give an estimate for future prediction once a method has been chosen. This also uses 25% of the data.

The split of the data is done randomly by a program to avoid any accidental bias, for each method the same subsets of data are used so that the methods can be compared fairly.

Regression Approach

The simplest classification method is using a linear regression model to predict the classification of the data. Taking each variable as a parameter in a proposed regression model, the equation $y = X\beta$ is fitted where β holds the values of the predicted parameters. Once the model has been fitted using the training data, the values of β are used to predict the classes of the validation data. If the predicted value is greater than or equal to 0.5, then the observation is classed as 1, if less than 0.5 then it is classed as 0. These predicted class labels are then compared to the actual validation data class labels to create a misclassification statistic.

K-Nearest Neighbours

The K-Nearest Neighbours method does not fit any sort of parametric model onto the data but instead classifies an observation by the classifications of the points nearest to it (its nearest neighbours). For a value of k and a data point x, this method looks at the classification of the k points nearest to x and classifies x to be the class that the k nearest points have in majority. For this dataset we calculated the distance between two points to be the Euclidean Distance and have chosen values of k from 1 to 50.

Discriminant Analysis

Discriminant Analysis describes a set of methods used to predict the class of an observation based on the measurements taken. In this report three methods of this type have been used, they are:

- **Canonical Variate Analysis (CVA)**, which tries to find a set of linear functions of the variables that can be used to define the optimal directions of separation among the classes.
- **Linear Discriminant Analysis (LDA)**, which is similar to CVA but assumes that each class is normally distributed with a mean and covariance matrix
- **Quadratic Discriminant Analysis (QDA)**, which is similar to LDA but assumes that the variances of each class are not equal therefore meaning that quadratic functions have to be fitted to the data instead of linear functions

Tree-based Methods

The final method involves classifying data using binary trees, which are structures made up by a series of nodes, leaves, splits and edges. The data is fitted onto the tree via a process called one-step look-ahead where at each stage, given the previous set of nodes and splits, the method tries to decide what the best variable and best cut-off point is for the next split. In this report binary splits have been used where a value is classed by either being less than or greater than or equal to a certain cut-off point t for a chosen variable.

Results

Using R the data was split into training, validation and test data subsets. Using the training and validation subsets each of the methods described above were applied to the data. The training data was used to create a classification rule prediction which was then applied to the validation data to give a set of predicted class labels. These predicted labels were then checked against the actual class labels from the validation data to determine if an observation had been classified correctly.

To compare the methods used, the misclassification rate is calculated, this equals the number of misclassified observations divided by the total number of observations.

Regression

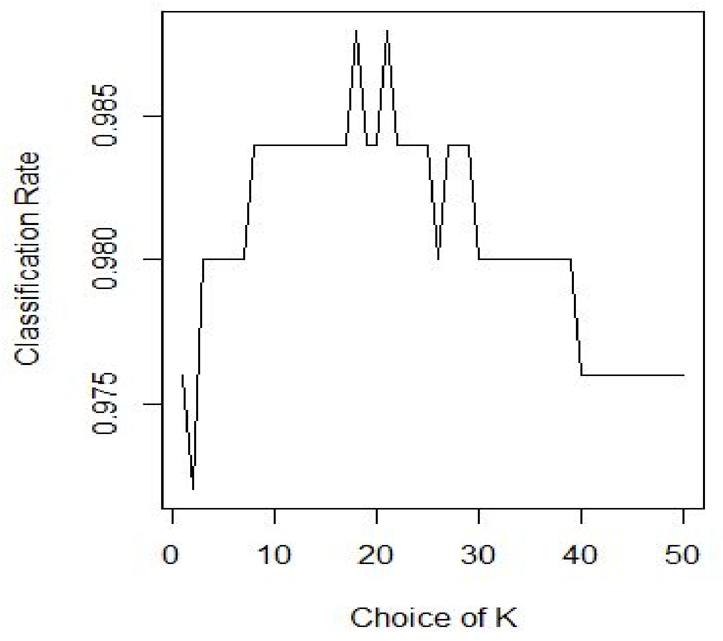
Fitting classification using a linear regression model gives the classification table shown below:

	Validation Data belonging to Class 0	Validation Data belonging to Class 1
Predicted Class 0	223	7
Predicted Class 1	0	20

The table shows that there were 7 values misclassified by this method (predicted to be noise when they were actually pulsars). There are no noise readings being interpreted as pulsars with the remaining observations classified correctly. For this method the misclassification rate is therefore 0.028

K-Nearest Neighbours

To choose the best value of k, all values of k from 1 to 50 were used to fit the k-nearest neighbours algorithm onto the data. The following graph of classification rates (1-misclassification rate) was then produced:



As seen from the graph, the classification rate spikes at two values of k ,18 and 21. As there are only two possible classes for data, an odd number of k is preferred (as an even number of k may result in a tie between class frequencies in the neighbours) therefore 21 is determined to be the best choice for k.

Fitting the k-nearest neighbours algorithm where k is equal to 21 gives the classification table shown below:

	Validation Data belonging to Class 0	Validation Data belonging to Class 1
Predicted Class 0	223	3
Predicted Class 1	0	24

This method improves on the regression approach by correctly classifying 4 of the misclassified values as pulsars. Once again there are misclassifications for noise as pulsars. The misclassification rate for this method is 0.012.

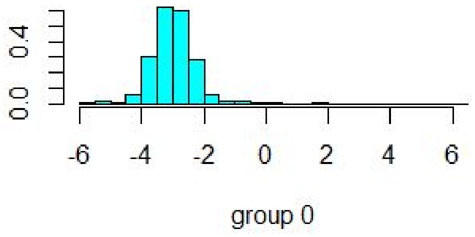
Discriminant Analysis

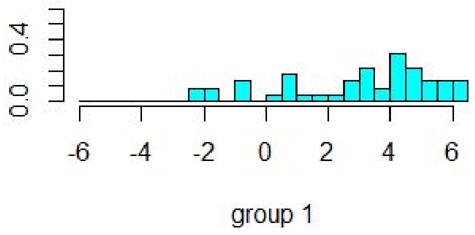
Canonical Variate Analysis (CVA): Carrying out CVA on the data produces the classification table:

	Validation Data belonging to Class 0	Validation Data belonging to Class 1
Predicted Class 0	223	0
Predicted Class 1	3	24

The classification table here is very similar to the one produced by k-nearest neighbours but now the 3 misclassified observations are now noise being interpreted as pulsars. The misclassification rate for this method is still 0.012.

When plotting CVA after it has been applied to the training data, the following histograms for each class can be created:





For group 0 (noise) a definite bell-shaped shape can be seen in the histogram therefore it can be assumed that this class follows a normal distribution. This bell-shape is still present in group 1 (pulsar) but is less obvious due to the large variance of the class. As a normal distribution can be assumed within each class, due to the presence of the bell-shaped histograms, LDA and QDA can be applied to the data successfully but due to the differing variances in the classes only QDA has been considered here.

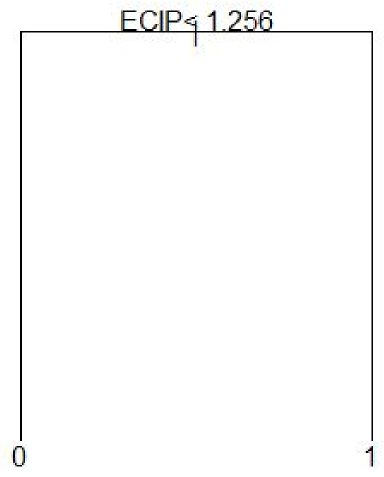
Quadratic Discriminant Analysis (QDA): When applying QDA to the data the following classification table can be produced.

	Validation Data belonging to Class 0	Validation Data belonging to Class 1
Predicted Class 0	221	2
Predicted Class 1	4	23

QDA produces a worse classification for the data than CVA or K-nearest neighbours but a slightly better classification than linear regression. QDA returns the lowest number of correctly classified noise observations compared to the other methods. The misclassification rate for this method is 0.024.

<u>Trees</u>

By application of the binary tree method, the following tree is created:



This tree classifies an observation by analyzing its ECIP (Excess Kurtosis of the Integrated Profile) value and classes the observation as noise if less than 1.256 and as a pulsar if otherwise. The following classification table in produced by this method:

	Validation Data belonging to Class 0	Validation Data belonging to Class 1
Predicted Class 0	222	1
Predicted Class 1	3	24

This method shows a slight improvement of the table created by QDA by correctly classifying two observations (one as noise, one as pulsar) but still does not improve on the classification provided by either CVA or K-nearest neighbours. The misclassification rate for this method is 0.016.

Conclusions

From assessing the misclassification rate for each method, the two best are K-nearest neighbours and CVA which each have the lowest misclassification rate of 0.012. Each method misclassified 3 observations where K-nearest neighbours misclassified some noise as pulsars with CVA doing the reverse for the same number of observations.

To determine which method would be preferred for the data set, both methods were ran again, using the test data instead of the validation data to calculate a misclassification rate for future values. For K-nearest neighbours the following table was produced:

	Validation Data belonging to Class 0	Validation Data belonging to Class 1
Predicted Class 0	226	5
Predicted Class 1	1	18

This gave a misclassification rate of 0.024, double the misclassification rate that was obtained using the validation data.

For CVA the following table was produced:

	Validation Data belonging to Class 0	Validation Data belonging to Class 1
Predicted Class 0	226	1
Predicted Class 1	7	16

This gave a misclassification rate of 0.032, which is larger than the misclassification rate of the K-nearest neighbors method. Therefore, it can be determined that the best method for classifying the data is the K-nearest neighbours method where k is equal 21, this method will misclassify data with a rate of 0.024 ie there is a 2.4% chance that data will be misclassified.

R Code Appendix

```
rm(list=ls())
load(“MM2017Project.RData")

data = HTRU.1

#######Split the Data into Test,Train and Validation##################
labels = data$Class
data = data[,-9]

n = nrow(data)
ind1 = sample(c(1:n),round(n/2))
ind2 = sample(c(1:n)[-ind1], round(n/4))
ind3 = setdiff(c(1:n), c(ind1,ind2))

train.data  = data[ind1,]
train.label = labels[ind1]

valid.data = data[ind2,]
valid.label = labels[ind2]

test.data = data[ind3,]
test.label = labels[ind3]

save(train.data,train.label,valid.data,valid.label,test.data,test.label, file = "data.txt")

########Data is saved, must be read in from now on##################

load(file = "data.txt")

######Regression########
res.lm = lm(train.label~.,data=train.data)
pred.class = predict(res.lm, valid.data)
pred.class = ifelse(pred.class >= 0.5, pred.class <- 1, pred.class <- 0)
corr.class.rate = sum(valid.label == pred.class)/nrow(valid.data)

table(pred.class, valid.label)

######Knn############
library(class)
corr.class.rate = rep(NA,50)
for(k in 1:50){
  pred = knn(train.data,valid.data,train.label,k=k)
  corr.class.rate[k] = sum(pred == valid.label)/nrow(valid.data)
}

corr.class.rate

plot(1:50,corr.class.rate, type = "l", xlab = "Choice of K", ylab = "Classification Rate")

pred.best = knn(train.data,valid.data,train.label,k=21)
table(pred.best, valid.label)

corr.class.rate

#####CVA###############
library(MASS)
train.lda = lda(train.data,train.label,prior = rep(0.5,2))
train.lda
plot(train.lda)

valid.lda = predict(train.lda,newdata = valid.data)
names(valid.lda)
valid.lda
xtab = table(valid.label,valid.lda$class)
xtab
```

```
1 - sum(diag(xtab))/sum(xtab)

##########QDA################
train.qda = qda(train.data,train.label)

valid.qda = predict(train.qda,valid.data)

xtab = table(valid.label,valid.qda$class)
xtab

1 - sum(diag(xtab))/sum(xtab)

#########Trees############

library(rpart)
train.tree = cbind(train.label,train.data)
valid.tree = cbind(valid.label,valid.data)

train.rp = rpart(train.label~., data = train.tree, method = "class")

train.rp
plot(train.rp,uniform = T, margin = 0.1)
text(train.rp)

valid.rp = predict(train.rp,valid.tree, type = "class")
xtab = table(valid.label,valid.rp)
xtab
1 - sum(diag(xtab))/sum(xtab)
```